SUPERVISED LEARNING

Online Shoppers Purchasing Intention

T09G94:

Joana Santos - up202006279
Mafalda Costa - up202006417
Mariana Carvalho - up202007620

AI – Assignment No. 2

PROBLEM

For this second IART assignment we are applying machine learning models and algorithms related to supervised learning. For our specific theme, **"Online Shoppers purchasing intention"**, we aim to predict whether an online shopper is likely to make a purchase or not, using **supervised machine learning algorithms for classification**. We will conduct an exploratory **data analysis** to determine the relevant features for our prediction task and **evaluate** various **models** to identify the most accurate and efficient one. Our goal is to provide valuable insights into customer behavior.

The input data we will use to make predictions and identify patterns consists in a data set of **17 features** of various types and **one target**, the **Revenue**, which indicates if a person made a buy or not. Seven of the features in the dataset are categorical, which require further pre-processing.

Some of the metrics that will be used to measure the performance of the algorithms are confusion matrix, performance during learning, precision, recall, accuracy and F1 measure; along with the time spent to train/test the models.

The models that we will use to predict the Revenue are Decision Tree, k-nearest neighbors, Random Forest, Support Vector Machines (SVM), eXtreme Gradient Boosting, Neural network and Naive Bayes.

REFERENCES

Related work that we're using to better understand and analyse the data:

- Sakar, C.O., Polat, S.O., Katircioglu, M. et al. Real-time prediction of online shoppers' purchasing intention using multilayer perceptron and LSTM recurrent neural networks. Neural Comput & Applic 31, 6893–6908 (2019). https://doi.org/10.1007/s00521-018-3523-0 https://link.springer.com/article/10.1007/s00521-018-3523-0
- Baati, K., Mohsil, M. (2020). Real-Time Prediction of Online Shoppers' Purchasing Intention Using Random Forest. In: Maglogiannis, I., Iliadis, L., Pimenidis, E. (eds) Artificial Intelligence Applications and Innovations. AIAI 2020. IFIP Advances in Information and Communication Technology, vol 583. Springer, Cham. https://doi.org/10.1007/978-3-030-49161-1_4 https://link.springer.com/chapter/10.1007/978-3-030-49161-1_4
- Deoraj, A. Predicting Customer Purchasing Itention. https://pats.cs.cf.ac.uk/@archive_file?p=1629&n=final&f=1-report.pdf&SIG=3b68283698dd64f4f82445abda026270536359bdd838a2d42592dcdf9fc03558
- https://www.kaggle.com/code/kankanj/online-shoppers-intention-prediction

ALGORITHMS, TOOLS AND FRAMEWORK

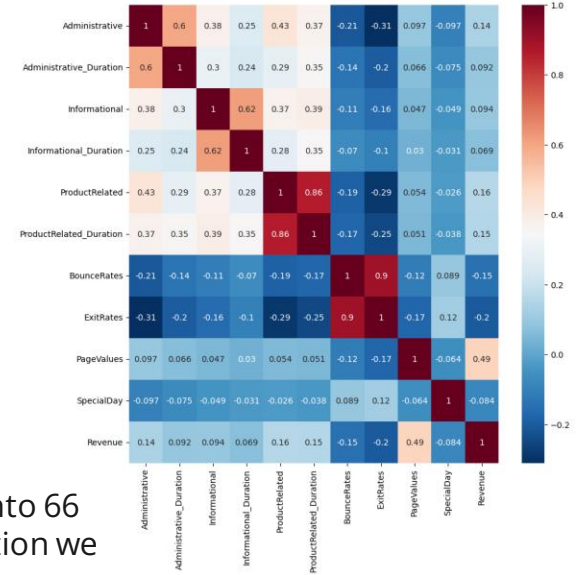
For this project we're using Jupyter Notebook, as it offers a simple way to write, visualize and explain the different steps. To process and visualize our data, we have incorporated several Python libraries, such as NumPy, Pandas, Matplotlib, Seaborn, Collections and imblearn . Additionally, we used OneHotEncoder from sklearn.preprocessing, to turn categorical attributes into numerical, and MinMaxScaler to scale the data.

Decision Tree	Through scikit-learn's DecisionTreeClassifier class.
k-nearest neighbors	Through scikit-learn's KNeighborsClassifier class.
Random Forest	Through scikit-learn's RandomForestClassifier class
Support Vector Machines (SVM)	Through scikit-learn's SVC class.
eXtreme Gradient Boosting	Through XGBoost
Neural network	Through scikit-learn's MLPClassifier
Naive Bayes	Through scikit-learn's GaussianNB class

DATA PRE-PROCESSING

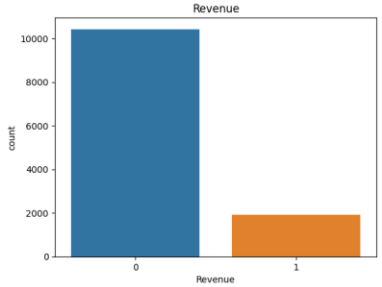
We began by analysing our data:

- zero **NULL values**;
- **duplicate rows**;
- **correlation between the numerical features**;
- **outliers**;
- identified 8 **categorical features**;
- the **target** (“Revenue”) had a very **imbalanced class distribution**;



We handled these issues with:

- **Removed the duplicate rows**;
- **Interquartille Range (IQR)** - to remove the outliers that are outside the 2nd percentile and 98th percentile.
- **OneHotEnconder** - transforms the categorical features into numerical representations, as a result, the 8 categorical features have been expanded into 66 attributes. (Less features will be used for the models due to the feature selection we implemented.);
- **MinMaxScaler** – used to normalize the range of features in a dataset, making them comparable;
- **SMOTE** and **RandomUnderSampler** to respectively perform **oversampling and undersampling** of the data;
- **Feature Selection** using **SelectKBest** with the **Mutual Information** as the scoring function – this optimizes our model's performance and reduces unnecessary complexity by identifying the features that have the most impact on the target;



MODELS AND RESULTS

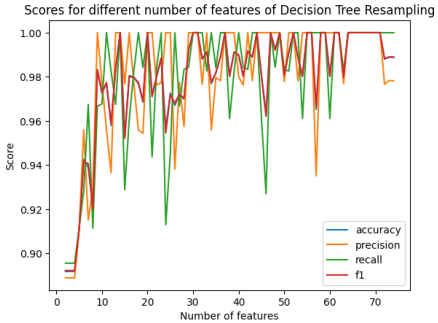
For every model, we firstly perform a **RandomizedSearchCV** in order to find the best range of values for the parameters used by the classifier. This way, we quickly reduce the possible parameters and therefore, efficiently perform a **GridSearchCV** that outputs the best ones to use for our classifiers. We also execute a **feature selection** for every model. It determines the features that have the most impact on the target, so the classifier only uses those, which reduces complexity and optimizes performance.

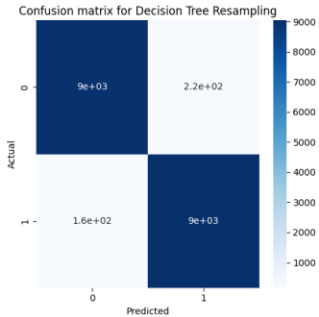
For each algorithm, we generated a plot of the ROC curves, showcasing the trade-off between true positive rate and false positive rate. We also measured metrics such as accuracy, precision, recall, and F1 for further analysis and comparison.

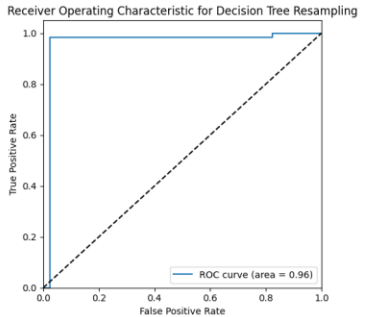
Decision Tree

Number of features used
- full data: 8
- resampled data: 14

	Model	Accuracy_score	Recall_score	Precision_score	F1_score
0	Decision Tree	0.897453	0.560193	0.714676	0.628074
1	Decision Tree Resampling	0.979511	0.982500	0.976661	0.979572





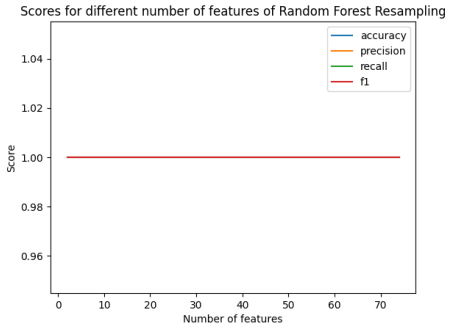


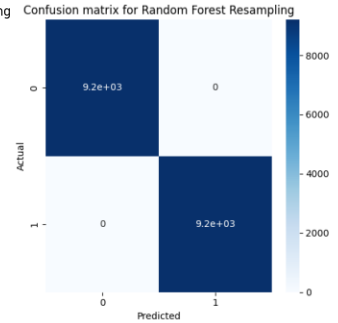
MODELS AND RESULTS

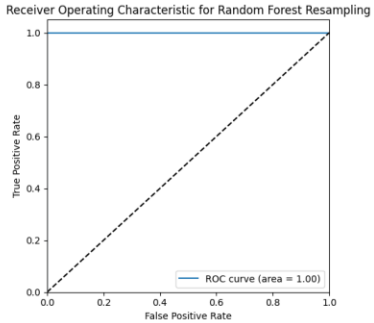
Random Forest

Number of features used
- full data: 5
- resampled data: 2

	Model	Accuracy_score	Recall_score	Precision_score	F1_score
0	Random Forest	0.895799	0.484216	0.753539	0.589577
1	Random Forest Resampling	1.000000	1.000000	1.000000	1.000000



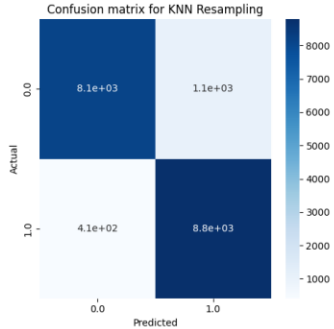


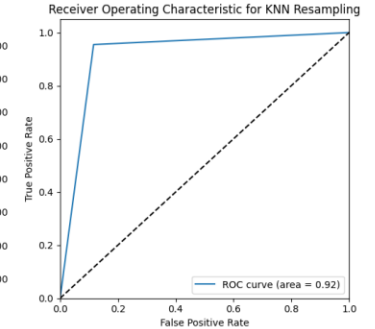


K-nearest neighbors (KNN)

Number of features used
- full data: 6
- resampled data: 40

	Model	Accuracy_score	Recall_score	Precision_score	F1_score
0	KNN	0.894145	0.498662	0.730980	0.592875
1	KNN Resampling	0.919565	0.955000	0.891799	0.922318



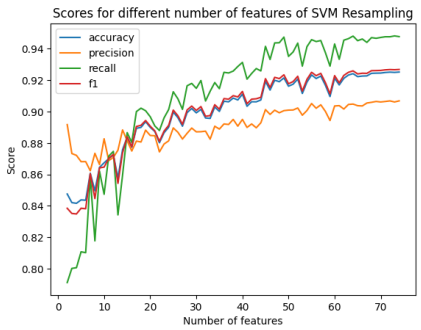


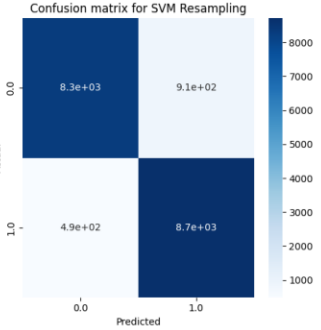
MODELS AND RESULTS

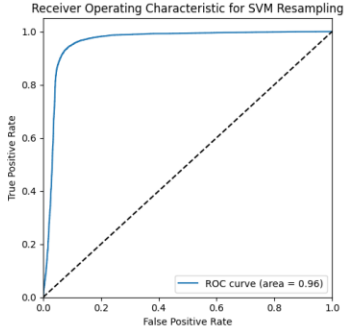
Support Vector Machines (SVM)

Number of features used
- full data: 7
- resampled data: 74

	Model	Accuracy_score	Recall_score	Precision_score	F1_score
0	SVM	0.886785	0.411450	0.740848	0.529068
1	SVM Resampling	0.924185	0.946848	0.905792	0.925865



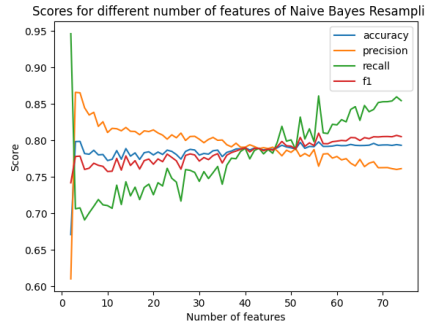


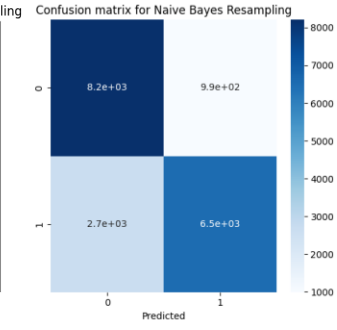


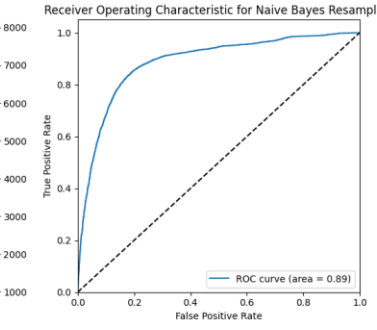
Naïve Bayes

Number of features used
- full data: 2
- resampled data: 4

	Model	Accuracy_score	Recall_score	Precision_score	F1_score
0	Naive Bayes	0.888604	0.447298	0.726957	0.553826
1	Naive Bayes Resampling	0.800815	0.709674	0.867872	0.780841





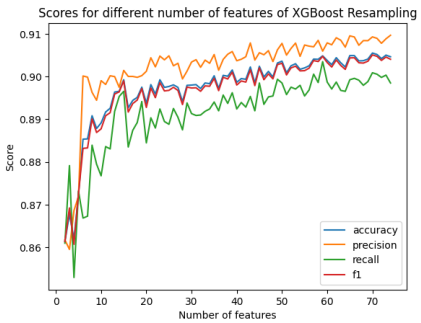


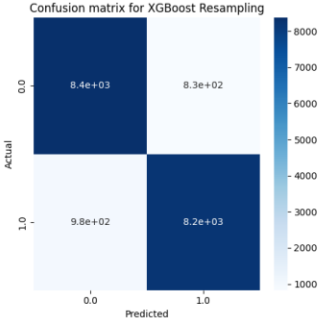
MODELS AND RESULTS

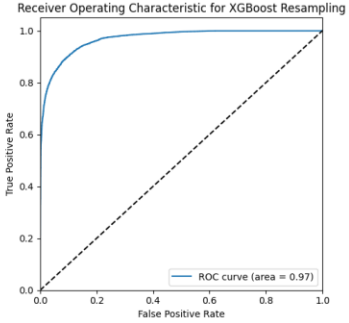
XGBoost

Number of features used
- full data: 41
- resampled data: 70

	Model	Accuracy_score	Recall_score	Precision_score	F1_score
0	XGBoost	0.901340	0.589085	0.721494	0.648601
1	XGBoost Resampling	0.901522	0.893043	0.908448	0.900680



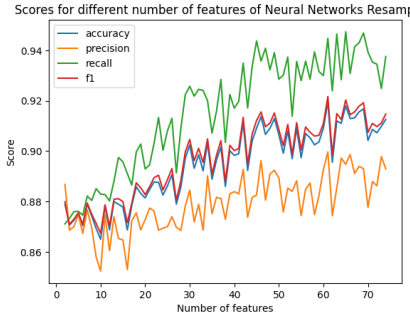


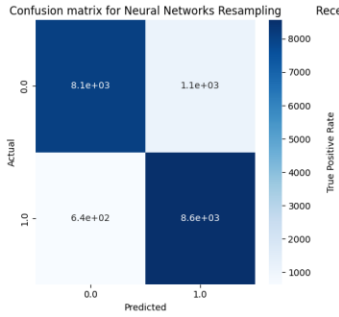


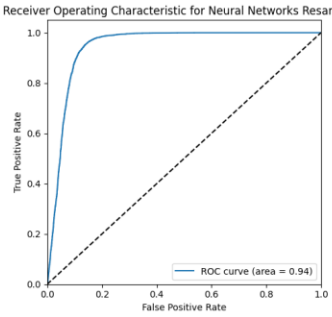
Neural Networks

Number of features used
- full data: 15
- resampled data: 61

	Model	Accuracy_score	Recall_score	Precision_score	F1_score
0	Neural Networks	0.900017	0.563403	0.728216	0.635294
1	Neural Networks Resampling	0.904022	0.930326	0.883829	0.906482

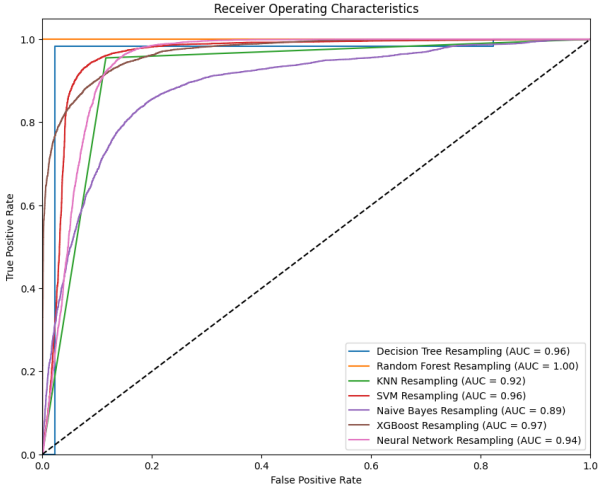






CONCLUSIONS

- **Decision Tree**: This model has seen a significant improvement with resampling. It now ranks second highest in all metrics. It appears to handle the balanced data well.
- **Random Forest**: This is the best performing model using the resampled data, with all accuracy, recall, f1 and precision with a score of 100%. This might seem ideal but could be a sign of overfitting. Random Forest also had the best AUC score, meaning it has a good overall performance in distinguishing between positive and negative instances.
- **KNN** and **SVM**: These models have seen a noticeable improvement as well, especially in recall. The F1 scores suggest a better balance of precision and recall after resampling. The AUC also increased on both models, especially on SVM, indicating really good performance in classifying the data.
- **Naive Bayes**: Although the accuracy diminished after resampling, the model's precision and recall improved significantly. The low values compared with the others might be the Gaussian distribution assumption, which does not fit all the features in our dataset.
- **XGBoost** and **Neural Networks**: Interestingly, these two models, which were previously top performers, didn't improve as much with resampling. While still having high scores, they did not outperform the Decision Tree, KNN, or SVM models in the resampled data. However, they still have a higher recall, precision, F1 score, and AUC than the imbalanced dataset.



	Model	Accuracy_score	Recall_score	Precision_score	F1_score
0	Decision Tree	0.897453	0.560193	0.714676	0.628074
1	Random Forest	0.895799	0.484216	0.753539	0.589577
2	KNN	0.894145	0.498662	0.730980	0.592875
3	SVM	0.886785	0.411450	0.740848	0.529068
4	Naive Bayes	0.888604	0.447298	0.726957	0.553826
5	XGBoost	0.901340	0.589085	0.721494	0.648601
6	Neural Networks	0.900017	0.563403	0.728216	0.635294

	Model	Accuracy_score	Recall_score	Precision_score	F1_score
0	Decision Tree Resampling	0.979511	0.982500	0.976661	0.979572
1	Random Forest Resampling	1.000000	1.000000	1.000000	1.000000
2	KNN Resampling	0.919565	0.955000	0.891799	0.922318
3	SVM Resampling	0.924185	0.946848	0.905792	0.925865
4	Naive Bayes Resampling	0.800815	0.709674	0.867872	0.780841
5	XGBoost Resampling	0.901522	0.893043	0.908448	0.900680
6	Neural Networks Resampling	0.904022	0.930326	0.883829	0.906482

☺