**Assignment 3**

**Tyler Trimble**

**Data and Data Processing:**

In this section explain your pre-processing, feature engineering, feature selection, and any data-augmentation you performed. Also, show a visualization of the data.

You must perform some data preprocessing, and you must explain what you did. Below is a list of things to consider. You must do some of these, but you don’t need to do all of them. Use your best judgement.

* What features will you use?
* How will you deal with categorical features?
* Will you eliminate uninformative features? If so, how will you determine which feature are uninformative?
* Will you perform any binning/bucketing?
* Will you perform any normalization or standardization?
* What is the class balance? Will you perform any under-sampling or oversampling?

**I did not use all features. I deemed that the booking\_ID feature did not carry any value in this context, so I dropped it from my feature list. There were three categorical features in the dataset: type\_of\_meal\_plan, room\_type\_reserved, and market\_segment\_type. I used one hot encoding for these features in order to make them numeric values. The other features I used as is. For the labels, I used mapping to make all cancelled rooms as ones in the series, and all non-cancelled rooms as zeros.**

You must state the number of samples, how you encoded the labels, and the number of true samples and false samples.

**There are 9726 total samples in the dataset. Of those 9726 samples, 3049 of them are true samples, meaning the rooms were cancelled. I encoded these true labels with the number 1. The other 6227 samples were false, or not cancelled, and were encoded with the number 0.**

You must visualize the data and insert a plot using PCA and another plot using TSNE. You must explain how much variability is explained in the PCA plot (per each dimension and combined). Your plots won’t have clear class boundaries. Explain what this tells you about the data.

sklearn.decomposition.PCA

sklearn.manifold.TSNE

Chart, scatter chart

Description automatically generated

**\*Because of CPU memory issues I was only able to plot 7500 samples out of the 9726 total\***

**Using pca.explained\_variance\_ratio, I am able to determine that the percentage of variance by each of the thirty components. The output explains the percentage of variance for each component in the principal component analysis, and they all add up to 1.0:**

**1: 1.14495144e-01. 2: 8.46248937e-02 3: 7.28471735e-02 4: 5.85682868e-02**

**5: 5.20810557e-02 6: 4.61550154e-02 7: 4.40664742e-02 8:4.35984853e-02**

**9: 3.74698798e-02 10: 3.52356531e-02 11: 3.50803839e-02 12: 3.37019108e-02**

**13: 3.30441754e-02 14: 3.22481463e-02 15: 3.17025094e-02 16: 2.97240135e-02**

**17: 2.83246101e-02 18: 2.75440167e-02 19: 2.65860614e-02 20: 2.63011274e-02**

**21: 2.11584376e-02 22: 1.98747449e-02 23: 1.70349381e-02 24: 1.53555993e-02**

**25: 1.37510498e-02 26: 1.01932456e-02 27: 9.23296824e-03 28: 6.80008054e-29**

**29: 1.77381388e-30 30: 3.44308764e-31**

**Results:**

You must fill in the following table

|  |  |  |
| --- | --- | --- |
| Model | Hyperparameters | Average 5-fold CV accuracy |
| Majority Class Baseline | Majority Class | 67.18% |
| KNN | n\_neighbors = 6 distance = euclidean | 77.59% |
| Decision Tree | max\_depth = 11 | 85.61% |
| Logistic Regression | alpha = 0.0001 | 76.02% |
| Polynomial Logistic Regression | alpha = 0.0001 | 73.87% |
| Neural Network | alpha = 0.001  hidden\_layer\_sizes = (10,) | 78.77% |
| Linear SVM | alpha = 0.0001 | 75.7% |
| RBF SVM | C = 10  gamma = 0.01 | 80.39% |

Using the following implementations:

Majority Class Baseline = code yourself

K Nearest Neighbors (KNN) = sklearn.neighbors.KNeighborsClassifier

Decision Tree = sklearn.tree. DecisionTreeClassifier

Logistic Regression = sklearn.linear\_model.SGDClassifier

Logistic Polynomial Regression = sklearn.linear\_model.SGDClassifier with sklearn.preprocessing.PolynomialFeatures

Neural Network = sklearn.neural\_network.MLPClassifier

Linear SVM = sklearn.linear\_model.SGDClassifier

RBF SVM = sklearn.svm.SVC

List any hyperparameters and the values that you specified in the hyperparameters column. List one hyperparameter=value pair per line, using the ones I listed for KNN as a template

Warning: Polynomial Logistic regression can be extremely slow for high degree polynomials, and RBF SVM can also be slow

**Best Results:**

In this section simply state the best performing classifier, its performance, and the hyperparameters you used.

**I found that the decision tree classifier gave me the best performance with an accuracy of 85.61% when evaluated on the test set. I evaluated depths of 1 to 20 and found that a max depth of 11 gave me the best results.**