**Week 3**

**Summary**

**Milestones achieved**

* Ran classifier model data set, and recorded the major metrics like: accuracy, TP rate, FP rate, precision, recall, F measure, the ROC area etc**.**
* Compared between different evaluation metrics.

**Conclusions**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Classifier** | **Data Set 1 (6 features)** | **Data Set 2 (15 features)** | **Data Set 3 (30 features)** | **Main Dataset** | **Selected\_features Dataset** |
| **Logistic Regression** | **Accuracy: 0.76, Precision: 0.76, Recall: 0.76, F1: 0.75, ROC AUC: 0.88** | **Accuracy: 0.79, Precision: 0.79, Recall: 0.79, F1: 0.79, ROC AUC: 0.92** | **Accuracy: 0.80, Precision: 0.79, Recall: 0.80, F1: 0.79, ROC AUC: 0.92** | **Accuracy: 0.82, Precision: 0.82, Recall: 0.82, F1: 0.82, ROC AUC: 0.94** | **Accuracy: 0.81, Precision: 0.81, Recall: 0.81, F1: 0.80, ROC AUC: 0.94** |
| **Decision Tree** | **Accuracy: 0.89, Precision: 0.89, Recall: 0.89, F1: 0.89, ROC AUC: 0.97** | **Accuracy: 0.88, Precision: 0.88, Recall: 0.88, F1: 0.88, ROC AUC: 0.89** | **Accuracy: 0.90, Precision: 0.90, Recall: 0.90, F1: 0.90, ROC AUC: 0.90** | **Accuracy: 0.91, Precision: 0.92, Recall: 0.91, F1: 0.91, ROC AUC: 0.91** | **Accuracy: 0.91, Precision: 0.91, Recall: 0.91, F1: 0.91, ROC AUC: 0.94** |
| **KNN Classifier** | **Accuracy: 0.87, Precision: 0.86, Recall: 0.87, F1: 0.87, ROC AUC: 0.94** | **Accuracy: 0.87, Precision: 0.86, Recall: 0.87, F1: 0.87, ROC AUC: 0.94** | **Accuracy: 0.90, Precision: 0.90, Recall: 0.90, F1: 0.90, ROC AUC: 0.96** | **Accuracy: 0.91, Precision: 0.91, Recall: 0.91, F1: 0.91, ROC AUC: 0.96** | **Accuracy: 0.92, Precision: 0.92, Recall: 0.92, F1: 0.92, ROC AUC: 0.98** |
| **Random Forest** | **Accuracy: 0.89, Precision: 0.89, Recall: 0.89, F1: 0.89, ROC AUC: 0.97** | **Accuracy: 0.89, Precision: 0.89, Recall: 0.89, F1: 0.89, ROC AUC: 0.97** | **Accuracy: 0.93, Precision: 0.93, Recall: 0.93, F1: 0.93, ROC AUC: 0.99** | **Accuracy: 0.95, Precision: 0.95, Recall: 0.95, F1: 0.95, ROC AUC: 0.99** | **Accuracy: 0.94, Precision: 0.94, Recall: 0.94, F1: 0.94, ROC AUC: 0.99** |

**Observations Across Classifiers and Datasets**

1. **Logistic Regression**:
   * We observed that performance improved as the number of features increased, peaking with the main dataset.
   * The highest ROC AUC for the selected features and main dataset (0.94) indicates strong class separation.
   * However, accuracy remained slightly lower than tree-based models, suggesting its limitations in handling complex feature interactions.
2. **Decision Tree**:
   * This model consistently performed well across all datasets.
   * High accuracy and ROC AUC scores demonstrate its flexibility in capturing both simple and complex patterns.
   * Minimal performance differences between selected features and the main dataset show that the tree was robust to feature selection.
3. **K-Nearest Neighbors (KNN)**:
   * Performance steadily improved as the number of features increased, with peak accuracy (0.92) and ROC AUC (0.98) for the selected features dataset.
   * Precision was slightly lower than Random Forest, which might indicate sensitivity to noise or feature scaling in higher-dimensional datasets.
4. **Random Forest**:
   * Random Forest outperformed all other classifiers, achieving the highest accuracy, F1-score, and ROC AUC across all datasets.
   * It performed exceptionally well on datasets with more features, reaching peak accuracy on the main dataset (0.95) with a near-perfect ROC AUC (0.99).
   * Its ensemble nature made it highly robust to overfitting.

**Comparing Datasets**

* **Dataset 1 (6 Features)**:
  + Simpler models like Logistic Regression performed adequately, but tree-based methods achieved better metrics.
  + The limited number of features constrained the models' ability to capture complex relationships.
* **Dataset 2 (15 Features)**:
  + Performance metrics improved moderately, especially for Decision Tree and KNN, highlighting the value of increased feature representation.
* **Dataset 3 (30 Features)**:
  + All classifiers showed their best performance here, suggesting that this dataset effectively captured significant variance.
  + Tree-based models demonstrated only marginal improvements, confirming their ability to manage large feature spaces.
* **Main Dataset**:
  + Random Forest and KNN excelled with the full feature set, leveraging the dataset's complexity.
  + Logistic Regression lagged behind, reflecting its limitations in handling non-linear relationships.
* **Selected Features Dataset**:
  + The performance was nearly identical to the main dataset, which suggests that our feature selection was effective.
  + Logistic Regression, Decision Tree, and Random Forest maintained strong results.

**Evaluation Metrics**

* **Accuracy**: This provided an overall trend, with Random Forest consistently achieving the highest scores.
* **Precision and Recall**: These metrics revealed trade-offs in identifying false positives and negatives. Random Forest showed the best balance.
* **F1-Score**: This confirmed overall classifier efficiency. Both KNN and Random Forest excelled here, especially on datasets with more features.
* **ROC AUC**: This metric demonstrated the ability of models to distinguish between classes. Random Forest achieved near-perfect scores (0.99), indicating its superiority.

**References**

* <https://medium.com/@outside2SDs/an-overview-of-correlation-measures-between-categorical-and-continuous-variables-4c7f85610365>
* <https://colab.research.google.com/drive/1mqfvb_NTjm1g_OPYrnck_EreWclqHzhx?usp=sharing>
* <https://www.tutorialspoint.com/machine_learning_with_python/index.htm>
* Müller, AC, & Guido, S 2016, Introduction to Machine Learning with Python : A Guide for Data Scientists, O'Reilly Media, Incorporated, Sebastopol. Available from: ProQuest Ebook Central. [29 September 2024].
* <https://ebookcentral.proquest.com/lib/hw/reader.action?docID=4698164&ppg=28>
* <https://github.com/ageron/handson-ml2/blob/master/03_classification.ipynb>

**Next Steps**

* Clustering Algorithms
* Organize the Pipeline; Modularize the code: Each step (downloading, loading, preprocessing, visualization, etc.) should be a function, making it easier to maintain and extend.
* Ensure the pipeline runs end-to-end: From downloading the dataset (if necessary) to splitting it into train and test sets.
* Working on image data set