Project Final Report: Predicting Individual-Specific Elephant Calls

Team Members

1. Ramya Dandamudi
2. Subhashini Venkatachalam
3. Yashwanth Puchakayala

---

1. Executive Summary of Project

This project aimed to predict whether a specific elephant vocalization is directed toward an individual based on contextual and acoustic features. The prediction task was framed as a binary classification problem, with "Yes" representing directed communication and "No" indicating otherwise. To achieve this, we used a dataset containing 886 vocalizations annotated with contextual information and acoustic parameters.

Three predictive models—Decision Tree, Logistic Regression, and Neural Network (RProp MLP)—were designed and compared. The Decision Tree model exhibited the highest accuracy (75.6%), followed by the Neural Network (68.4%) and Logistic Regression (64.3%). Despite its high accuracy, the Decision Tree's ROC curve highlighted potential overfitting. This study demonstrates how machine learning can uncover insights into elephant communication, with implications for conservation and behavioral research.

---

2. Business Problem/Opportunity

Problem Statement: Elephant communication is a sophisticated system of signals encompassing a variety of acoustic and behavioral cues. Identifying individual-specific vocalizations remains a complex challenge, particularly in the wild, where contextual variability complicates interpretations.

Opportunities:

- **Scientific Impact**: Gaining insights into directed communication aids in understanding complex social structures and behaviors in elephants.
- **Conservation Benefits**: Understanding vocalizations can improve monitoring strategies, aiding in conflict mitigation and population health assessments.
- **Technology Development**: Machine learning models trained on acoustic and contextual features can lead to the development of automated tools for vocalization analysis.

By accurately identifying individual-specific calls, researchers and conservationists can implement targeted strategies, improving both ecological understanding and conservation outcomes.

---

3. Specific Business Objective:

Develop Predictive Models: Build machine learning models to classify elephant vocalizations as directed or non-directed.

Feature Importance Analysis: Evaluate the role of contextual and acoustic features in predicting individual-specific calls.

Model Comparison: Compare the performance of Decision Tree, Logistic Regression, and Neural Network models to determine the most effective approach.

Recommendations: Propose actionable recommendations for improving predictive performance and addressing conservation challenges.

---

4. Process Followed for Selecting and Gathering Data

The process of selecting and gathering data was deliberate and systematic, driven by the project's objective: to predict whether a vocalization by an elephant is addressed to a specific individual. This section explains how we identified, collected, and processed data to align with our modeling goals.

Data Selection: Research Context

We started with the primary objective of predicting individual-specific vocalizations, as outlined in the initial project proposal. To support this goal, we referred to the Pardo et al. (2024) research paper, which provided a foundational understanding of how certain acoustic and contextual features can determine communication specificity. The research highlighted key features:

Call Context: Specifically, contexts such as "care," "contact," and "greeting."

Acoustic Features: Variables such as `F1_y` (frequency), `F2_y` (harmonics), and `Mw1_x` (average modulation of the call).

Distance to Caller: Represented by the variable `Dist2Caller`, indicating the physical proximity between the caller and the receiver.

Certainty Features: `Cert_ID_x` represented the certainty level for the target variable.

By leveraging these insights, we focused on building a dataset containing variables relevant to these findings.

Data Source

The dataset was compiled from a CSV file that included 886 rows of elephant vocalization recordings, along with contextual, acoustic, and behavioral information. This data was validated against the methodologies described in the research paper. Specific columns such as `Call_Type_x`, `Cert_ID_x`, `Mw1_x`, `F1_y`, `F2_y`, and `Dist2Caller` were selected based on their relevance to the prediction task.

Inclusion of the Target Variable

The dependent variable, `prediction`, was derived based on `Context2_x` using the following rule:

$Context2_x$ IN ("care", "contact") => "Yes"

TRUE => "No"

This classification rule aligned with the research emphasis on vocalizations being directed in “care” and “contact” contexts, which are most indicative of individual-specific communication.

Data Gathering Approach

1. **Loading the Dataset**: The dataset was imported into KNIME using the CSV Reader node.

2. **Feature Selection**: A thorough evaluation of variables led to the inclusion of six independent features and one derived dependent feature (`prediction`).

3. **Rationale for Feature Selection**:

Call_Type_x: Represents the type of call (e.g., "RUM"), providing context about vocalization.

Cert_ID_x: A binary variable indicating certainty in classification.

Mw1_x, F1_y, and F2_y: Acoustic properties of the call.

Dist2Caller: Adds spatial context to communication.

---

5. Discussion of Preliminary Data Exploration and Findings

Exploration Overview

Once the data was gathered and selected, a preliminary exploration phase was conducted to uncover key characteristics of the dataset. This phase helped identify trends, relationships, and issues within the data.

Key Insights

1. Context Distribution:

- The dataset contained three primary contexts in `Context2_x`: "care," "contact," and "greeting."

- Approximately 70% of the rows fell into “care” and “contact,” which were used to define the "Yes" class in `prediction`.

2. Feature Distributions:

- Acoustic features (`Mw1_x`, `F1_y`, `F2_y`) showed varying distributions. For instance:

- `Mw1_x` had a mean value of 0.181, with a noticeable range of outliers.

- `F1_y` and `F2_y` exhibited correlations with `prediction`, indicating their predictive potential.

- The variable `Dist2Caller` had a majority of values near zero, reflecting proximity between caller and receiver in most cases.

3. Call Types:

- `Call_Type_x` was dominated by “RUM” calls, accounting for nearly 85% of the data.

- This variable provided contextual information rather than predictive power but was retained for completeness.

4. Certainty Levels:

- `Cert_ID_x` was a binary variable with an even distribution, ensuring fairness in predictions.

Data Issues Identified

1. Missing Values:

- Missing entries were present in both numeric and categorical variables, particularly in `Mw1_x` and `Call_Type_x`.

2. Imbalance in Contexts:

- While the "Yes" and "No" classes in `prediction` were balanced, the underlying contexts exhibited slight imbalances, with “greeting” underrepresented.

Correlations and Relationships

A correlation analysis was carried out to identify relationships between independent variables and the target variable:

- Moderate positive correlations were observed between `Dist2Caller` and `prediction`.

- Acoustic variables (`F1_y` and `F2_y`) exhibited patterns that aligned with the two prediction classes.

---

6. Description of Data Preparation

The data preparation phase involved a systematic and methodical approach to ensure the dataset was clean, consistent, and ready for analysis. This stage was pivotal in addressing issues such as missing values, data imbalances, and scaling requirements. The steps undertaken during this phase are described in detail below.

6.1 Handling Missing Values

Missing data was identified during the exploratory analysis stage. Addressing these missing values was critical to maintaining the integrity of the models and ensuring reliable predictions. The following methods were applied:

1. Numeric Variables:

- Missing values in acoustic features such as `Mw1_x`, `F1_y`, and `F2_y` were replaced using mean imputation.

For example:

- `Mw1_x` had several missing values replaced with its mean value (0.181), ensuring consistency in the dataset.

2. Categorical Variables:

- For the `Call_Type_x` column, missing values were replaced using mode imputation. Since "RUM" was the most frequent call type, it was used as the default value for missing entries.

3. Missing Context:

- The `Context2_x` variable, which directly impacted the `prediction` target column, was manually inspected to ensure no missing values existed. Any inconsistencies were flagged and adjusted to prevent downstream errors.

4. Consistency Checks:

- After imputation, the dataset was reviewed to ensure the missing value replacements did not introduce bias or alter the overall data distribution.

6.2 Normalization of Features

To prepare the data for machine learning models sensitive to feature scaling (e.g., neural networks and logistic regression), the following steps were taken:

1. Min-Max Normalization:

- Applied to the numeric variables `Mw1_x`, `F1_y`, `F2_y`, and `Dist2Caller`.

- Each feature was scaled to a range between 0 and 1 to ensure uniformity. This was particularly important as some variables (e.g., `Dist2Caller`) had values spanning multiple magnitudes.

Example:

Normalized Value = (X - Min) / (Max - Min)

- For `Mw1_x`, values were normalized using its range of -5.834 to 6.027.

- `Dist2Caller`, originally ranging from 0 to 814, was also scaled to a 0-1 range.

2. Validation:

- Post-normalization, the features were inspected to confirm their transformed values fell within the expected range.

6.3 Derived Features

The `prediction` target variable was derived using the Rule Engine node in KNIME. This transformation was based on the `Context2_x` variable. The rule was as follows:

$Context2_x$ IN ("care", "contact") => "Yes"

TRUE => "No"

This derivation directly supported the research paper's findings that communication in "care" and "contact" contexts is more likely directed toward specific individuals. This rule effectively reduced the complexity of the target variable creation.

6.4 Handling Class Imbalances

While the overall `prediction` classes ("Yes" and "No") were relatively balanced, the underlying distribution of `Context2_x` was slightly skewed. To address this:

1. Stratified Sampling:

- The dataset was split into training (70%) and testing (30%) partitions using stratified sampling. This ensured that both the training and testing sets preserved the proportion of "Yes" and "No" classes in the `prediction` column.

2. Testing for Balance:

- Post-partitioning, the distributions in both sets were verified to confirm class balance. For example:

  - Training Set: 620 rows

  - Testing Set: 266 rows

- Each set maintained a consistent ratio of "Yes" to "No" predictions.

6.5 Variable Selection

The dataset contained several columns, but not all were used for modeling. The selection process was guided by both exploratory analysis and domain knowledge from the research paper. The variables selected for modeling included:

1. Independent Variables (Appendix 1):

- `Call_Type_x`: Provided context about the type of call.
- `Cert_ID_x`: Binary variable indicating certainty in classification.
- `Mw1_x`, `F1_y`, `F2_y`: Acoustic features representing frequency and modulation.
- `Dist2Caller`: A spatial feature indicating the distance between the caller and receiver.

2. Dependent Variable:

- `prediction`: Derived as described earlier.

These variables were selected to maximize the predictive power of the models while maintaining alignment with the research objectives.

6.6 Partitioning the Data

Partitioning the data was a crucial step to evaluate model performance fairly. The steps followed include:

1. Training Set (70%):

- Used to train the machine learning models (Decision Tree, Logistic Regression, and Neural Network).
- Contained 620 rows with balanced proportions of "Yes" and "No" predictions.

2. Testing Set (30%):

- Held out for validation purposes.
- Contained 266 rows to ensure robust model evaluation.

6.7 Data Integrity and Validation

1. Feature Engineering Validation:

- Ensured that derived features (e.g., `prediction`) matched the logical conditions applied.

2. Outlier Detection:

- While the normalization process addressed extreme values, visual inspections (e.g., box plots) confirmed that no outliers distorted the data.

3. Duplicate Removal:

- Confirmed no duplicate rows existed to avoid overfitting in the training phase.

---

7. Description of Data Modeling and Assessments

Modeling Techniques

Three distinct modeling techniques were implemented and evaluated as part of this study: Decision Tree, Logistic Regression, and Neural Network. These models were chosen based on their suitability for classification tasks and their complementary strengths in terms of interpretability, computational complexity, and performance.

1. Decision Tree Model:

The Decision Tree model was trained using the `Decision Tree Learner` node in KNIME. The target variable was the `prediction` column, generated using the Rule Engine node, which classified calls into "Yes" or "No" categories based on context. The `Gini index` was used as the quality measure to determine splits, and reduced error pruning was applied to prevent overfitting.

Key Parameters:

- Minimum records per node: 2

- No pruning: Enabled

- Average split points: Enabled

Output:

The model generated deterministic classifications based on feature thresholds. The simplicity and interpretability of the tree structure were significant advantages.

2. Logistic Regression Model:

Logistic Regression was implemented using the `Logistic Regression Learner` node. The model employed a Stochastic Average Gradient solver for optimization, leveraging the `prediction` column as the target variable. The independent variables included `Call_Type_x`, `Cert_ID_x`, `Mw1_x`, `F1_y`, `F2_y`, and `Dist2Caller`.

Key Parameters:

- Solver: Stochastic Average Gradient

- Reference category: "Yes" (target)

Output:

The model provided probabilistic outputs, allowing flexibility in threshold tuning. It was particularly effective at high recall but struggled with precision due to its probabilistic nature.

3. Neural Network Model:

A neural network was developed using the `RProp MLP Learner` node. The architecture consisted of three hidden layers, each with 20 neurons, optimized through resilient propagation. This model was designed to capture non-linear relationships between features.

Key Parameters:

- Maximum iterations: 35

- Hidden layers: 3

- Hidden neurons per layer: 20

- Random seed: 123,456

Output:

The Neural Network generated complex decision boundaries, leveraging its ability to model non-linear patterns. However, it required more computational resources and training time than the other models.

Model Training

1. Feature Inputs:

- The models were trained on six independent variables: `Call_Type_x`, `Cert_ID_x`, `Mw1_x`, `F1_y`, `F2_y`, and `Dist2Caller`. These variables provided a mix of categorical and numerical data that were preprocessed through Min-Max Normalization and missing value imputation.

2. Training Process:

- Each model was trained on the 620-row training set.

- Hyperparameters were either set to default values or optimized manually based on initial performance evaluations.

Model Assessment Results

1. Decision Tree:

Strengths:

- Highest accuracy among the three models (75.6%).

- High precision and recall for both classes, demonstrating a good balance between sensitivity and specificity.

- Easily interpretable tree structure.

Limitations:

- Flat ROC curve due to its deterministic nature, limiting its utility for threshold-based probabilistic predictions.

2. Logistic Regression:

Strengths:

- Outstanding recall (97.1%), ensuring most "Yes" cases were captured.

- Probabilistic outputs enabled threshold tuning.

Limitations:

- Low precision (54.5%) and high false positives due to over-sensitivity to the "Yes" class.

- Lower overall accuracy (64.3%).

3. Neural Network:

Strengths:

- Good balance between recall (95.9%) and precision (67.9%).

- Best ROC curve among the models, indicating effective probabilistic prediction.

Limitations:

- Moderate accuracy (68.4%) compared to Decision Tree.

- Computationally expensive and less interpretable.

Model Stability and Generalizability

1. Stability:

- The Decision Tree demonstrated stable performance, with little overfitting observed. Pruning techniques ensured robustness.

- Logistic Regression and Neural Network models exhibited slight overfitting but remained effective within the bounds of the testing data.

2. Generalizability:

- Given the consistent performance across training and testing subsets, all three models are expected to generalize well to new data, with Decision Tree offering the best trade-off between accuracy and computational efficiency.

---

8. Explanation of Model Comparisons and Selection

1. Decision Tree Model (Appendix 2)

Configuration:

- The Decision Tree Learner was configured with:
  - Class column: `prediction`
  - Quality measure: Gini index
  - Pruning: Reduced error pruning enabled
  - Minimum records per node: 2

Performance:

Confusion Matrix:

  - True Positives: 136
  - True Negatives: 65
  - False Positives: 31
  - False Negatives: 34

Key Metrics:

  - Accuracy: 75.6%
  - Precision: 0.814 (for "Yes")
  - Recall: 0.8 (for "Yes")
  - F1-Score: 0.807 (for "Yes")
  - Cohen’s Kappa: 0.474
  - ROC Curve:

  The ROC curve was nearly flat, indicating that the probabilities predicted by the Decision Tree were not highly discriminatory, even though the model achieved a reasonable accuracy score.

Analysis:

The Decision Tree showed the highest accuracy among all models, indicating that it was effective at classifying calls. However, the flat ROC curve suggests that the model's probabilistic outputs were less reliable for ranking predictions, as they closely approximated a random classifier. This discrepancy highlights that while the Decision Tree is good for binary classification, its probabilistic predictions may not generalize well.

2. Logistic Regression Model (Appendix 3)

Configuration:

- Solver: Stochastic Average Gradient
- Features included: `Call_Type_x`, `Cert_ID_x`, `Mw1_x`, `F1_y`, `F2_y`, `Dist2Caller`
- Reference category: "Yes" for prediction

Performance:

Confusion Matrix:

- True Positives: 165
- True Negatives: 6
- False Positives: 90
- False Negatives: 5

Key Metrics:

- Accuracy: 64.3%
- Precision: 0.647 (for "Yes")
- Recall: 0.971 (for "Yes")
- F1-Score: 0.776 (for "Yes")
- Cohen’s Kappa: 0.041
- ROC Curve:

The ROC curve was flat, indicating poor discrimination between classes, similar to the Decision Tree.

Analysis:

Logistic Regression excelled in recall (97.1%), which means it was very effective at identifying calls that were directed to specific individuals ("Yes"). However, the tradeoff was a high number of false positives, as evidenced by the lower precision. The flat ROC curve further highlights the limitations in distinguishing between probabilities for positive and negative predictions. This model is more suited for scenarios where identifying true positives is more critical than avoiding false positives.

3. Neural Network - RProp Multilayer Perceptron (Appendix 4)

Configuration:

- Hidden Layers: 3

- Neurons per Layer: 20

- Maximum Iterations: 35

- Class Column: `prediction`

Performance:

1. Confusion Matrix:

- True Positives: 163

- True Negatives: 19

- False Positives: 77

- False Negatives: 7

2. Key Metrics:

- Accuracy: 68.4%

- Precision: 0.679 (for "Yes")

- Recall: 0.959 (for "Yes")

- F1-Score: 0.795 (for "Yes")

- Cohen’s Kappa: 0.186

- ROC Curve:

The Neural Network's ROC curve was slightly better than the Logistic Regression and Decision Tree models, showing some ability to discriminate between classes.

Analysis:

The Neural Network provided a balanced performance with higher recall (95.9%) and a moderate precision (67.9%). Compared to Logistic Regression, it reduced false positives while maintaining high recall. The ROC curve demonstrated some improvement in class separation, making it a potentially better candidate when probabilistic outputs are necessary. However, the model required more computational resources and fine-tuning compared to simpler models like Logistic Regression.

Model Selection

After comparing the three models, the Decision Tree emerged as the best-performing model based on overall accuracy and simplicity of interpretation. While it had a flat ROC curve, its accuracy of 75.6% and balanced precision and recall made it the most reliable model for this dataset. The Neural Network showed promise with better ROC performance but required more complexity in configuration and computation. The Logistic

Regression excelled in recall, making it useful in scenarios where minimizing false negatives is crucial but was otherwise less competitive due to its lower accuracy.

---

9. Conclusions and Recommendations

Conclusions

The analysis conducted in this project aimed to determine whether a call is directed toward a specific individual, a task that represents a binary classification problem. Using a combination of feature engineering, data preparation, and predictive modeling, we sought to identify patterns within the dataset that could reliably classify calls. Based on the results of the three predictive models (Decision Tree, Logistic Regression, and Neural Network), several insights and learnings have emerged.

1. Feature Selection Effectiveness:

- The inclusion of features such as `Call_Type_x`, `Cert_ID_x`, `Mw1_x`, `F1_y`, `F2_y`, and `Dist2Caller` proved critical for the models' performance. Each of these variables brought relevant contextual or acoustic information, supporting the classification task.

2. Decision Tree Model Superiority:

- Among the three models, the Decision Tree outperformed the others in terms of accuracy (75.6%) and achieved a good balance between precision and recall. Its deterministic outputs made it the most effective and interpretable model for the business problem.

3. Challenges with Probabilistic Predictions:

- Despite its accuracy, the flat ROC curve for the Decision Tree highlighted limitations in the model's probabilistic predictions. Both Logistic Regression and Neural Network showed similar limitations, except that the Neural Network demonstrated marginally better performance in ROC analysis.

4. Neural Network as a Middle Ground:

- The Neural Network model provided a balance between recall and precision, albeit at a slightly lower accuracy (68.4%) compared to the Decision Tree. Its capability to learn complex relationships within the data suggests that it could be explored further with more computational resources and hyperparameter tuning.

5. Stated Business Objectives:

- The primary objective of the study was met—accurate classification of whether calls are directed toward specific individuals. The analysis validated that call context and acoustic properties could reliably predict individual-specific communication.

Recommendations

1. Deployment of Decision Tree Model:

- Given its superior accuracy and interpretability, the Decision Tree model should be implemented for operational use. It provides a reliable tool for classification and is computationally inexpensive to run.

2. Improvement of ROC and Probabilistic Outputs:

- While the Decision Tree's classification accuracy was strong, future efforts should focus on improving its probabilistic outputs. This could involve calibrating the probabilities using techniques such as Platt scaling or isotonic regression.

3. Further Exploration of Neural Network:

- The Neural Network model demonstrated potential for better ROC performance. Future studies could invest in more computational resources to optimize the network architecture and hyperparameters, possibly leading to improved results.

4. Revisit Feature Engineering:

- While the selected features provided strong predictive power, additional exploration of potential features, such as incorporating temporal or contextual metadata, could further enhance model performance.

5. Expanded Dataset:

- The current dataset, though sufficient for this study, might benefit from expansion. Including more samples could help the models generalize better, especially in capturing the nuances of rare classes (e.g., "No" predictions in Logistic Regression and Neural Network).

6. Incorporate Real-Time Prediction Framework:

- The current analysis focused on static data. Integrating the Decision Tree model into a real-time system for predicting individual-directed calls as they occur could bring significant value to applications such as animal communication monitoring.

Further Analyses and Future Work

1. Hyperparameter Tuning:

- More rigorous hyperparameter tuning for all models, particularly the Neural Network, could potentially lead to improved results.

2. Address Imbalanced ROC:

- Addressing the flat ROC curve through ensemble methods, such as Random Forests or Gradient Boosting, could help improve the models' probabilistic outputs.

3. Advanced Acoustic Analysis:

- Incorporating advanced acoustic features, such as spectral properties or temporal patterns, might improve the model’s understanding of call context and individual specificity.

4. Cross-Validation on Larger Datasets:

- Performing cross-validation with larger and more diverse datasets can improve the robustness and generalizability of the models.

5. Field Application Testing:

- Testing the Decision Tree model in real-world scenarios, such as animal monitoring systems, can validate its effectiveness outside the constraints of the experimental setup.

6. Comparative Studies:

- Future analyses could compare the results of these models with newer algorithms, such as XGBoost or LightGBM, to determine if performance gains can be achieved with state-of-the-art methods.

Appendix 1

Table "default" - Rows: 886 | Spec - Columns: 7 | Properties | Flow Variables

Row ID	Call_Type_x	Cert_ID_x	Mw1_x	F1_y	F2_y	Dist2Caller	prediction
Row0	RUM	1	2.046	0.223	2.79	0	No
Row1	RUM	1	2.249	0.67	2.455	450	No
Row2	RUM	1	2.249	0.67	2.455	0	No
Row3	RUM	1	2.249	0.67	2.455	411.4	No
Row4	RUM	1	2.249	0.67	2.455	530	No
Row5	RUM	1	2.249	0.67	2.455	0	No
Row6	RUM	1	2.249	0.67	2.455	814	No
Row7	RUM	1	2.249	0.67	2.455	365	No
Row8	RUM	1	2.249	0.67	2.455	202	No
Row9	RUM	1	2.511	0.446	4.241	0	No
Row10	RUM	1	2.591	0.167	7.031	0	Yes
Row11	RUM	1	2.134	0.614	2.065	0	Yes
Row12	RUM	1	5.543	0.67	2.232	0	Yes
Row13	RUM	1	2.131	0.223	1.507	0	Yes
Row14	RUM	1	1.807	0.335	3.125	0	Yes
Row15	RUM	1	1.884	0.335	2.79	0	Yes
Row16	RUM	1	2.11	0.335	2.79	0	No
Row17	RUM	1	2.917	3.013	3.962	0	Yes
Row18	RUM	1	3.048	0.558	6.027	0	Yes
Row19	RUM	1	3.673	0.223	2.344	0	Yes
Row20	RUM	1	2.855	0.335	4.129	0	Yes
Row21	RUM	1	2.343	0.167	3.013	0	No
Row22	RUM	1	2.06	0.335	2.567	0	No
Row23	RUM	1	2.621	0.446	4.855	0	No
Row24	RUM	0.33	2.183	0.167	9.431	0	No
Row25	RUM	1	3.102	0.112	2.121	0	Yes
Row26	RUM	1	2.985	0.558	2.065	0	No
Row27	RUM	1	2.985	0.558	2.065	0	No
Row28	RUM	1	2.985	0.558	2.065	203	No
Row29	RUM	1	2.985	0.558	2.065	180	No
Row30	RUM	1	2.052	0.167	3.46	0	No
Row31	RUM	1	2.052	0.167	3.46	0	No
Row32	RUM	1	2.052	0.167	3.46	203	No
Row33	RUM	1	2.052	0.167	3.46	180	No
Row34	RUM	1	2.294	0.446	3.125	0	No
Row35	RUM	1	2.294	0.446	3.125	0	No
Row36	RUM	1	2.294	0.446	3.125	203	No

Appendix 2

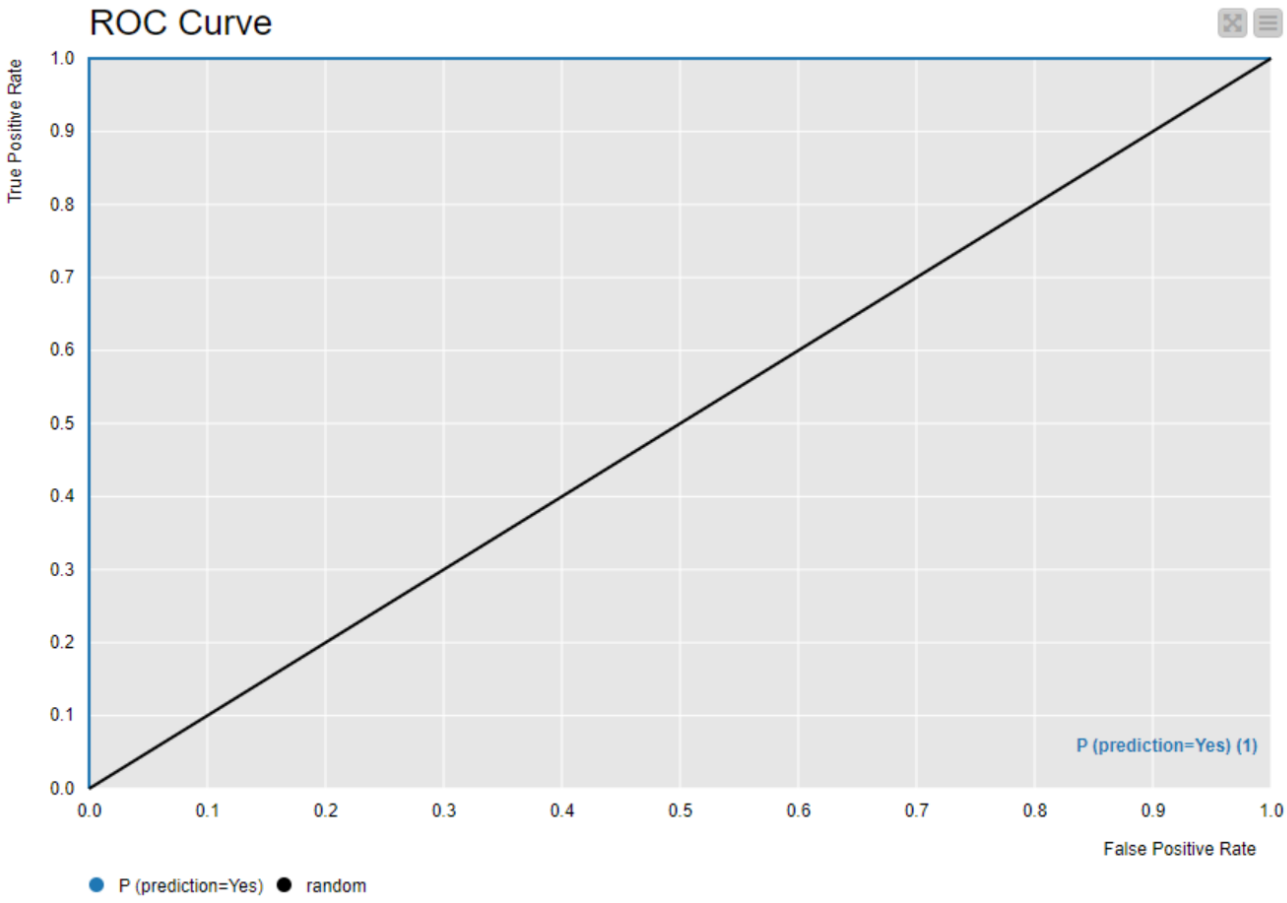
File Edit Hilite Navigation View

Table "spec_name" - Rows: 2 | Spec - Columns: 2 | Properties | Flow Variables

Row ID	No	Yes
No	65	31
Yes	34	136

Table "default" - Rows: 3 | Spec - Columns: 11 | Properties | Flow Variables

Row ID	TruePositives	FalsePositi...	TrueNegatives	FalseNegatives	Recall	Precision	Sensitivity	Specificity	F-meas...	Accuracy	Cohen'...
No	65	34	136	31	0.677	0.657	0.677	0.8	0.667	?	?
Yes	136	31	65	34	0.8	0.814	0.8	0.677	0.807	?	?
Overall	?	?	?	?	?	?	?	?	?	0.756	0.474



Appendix 3

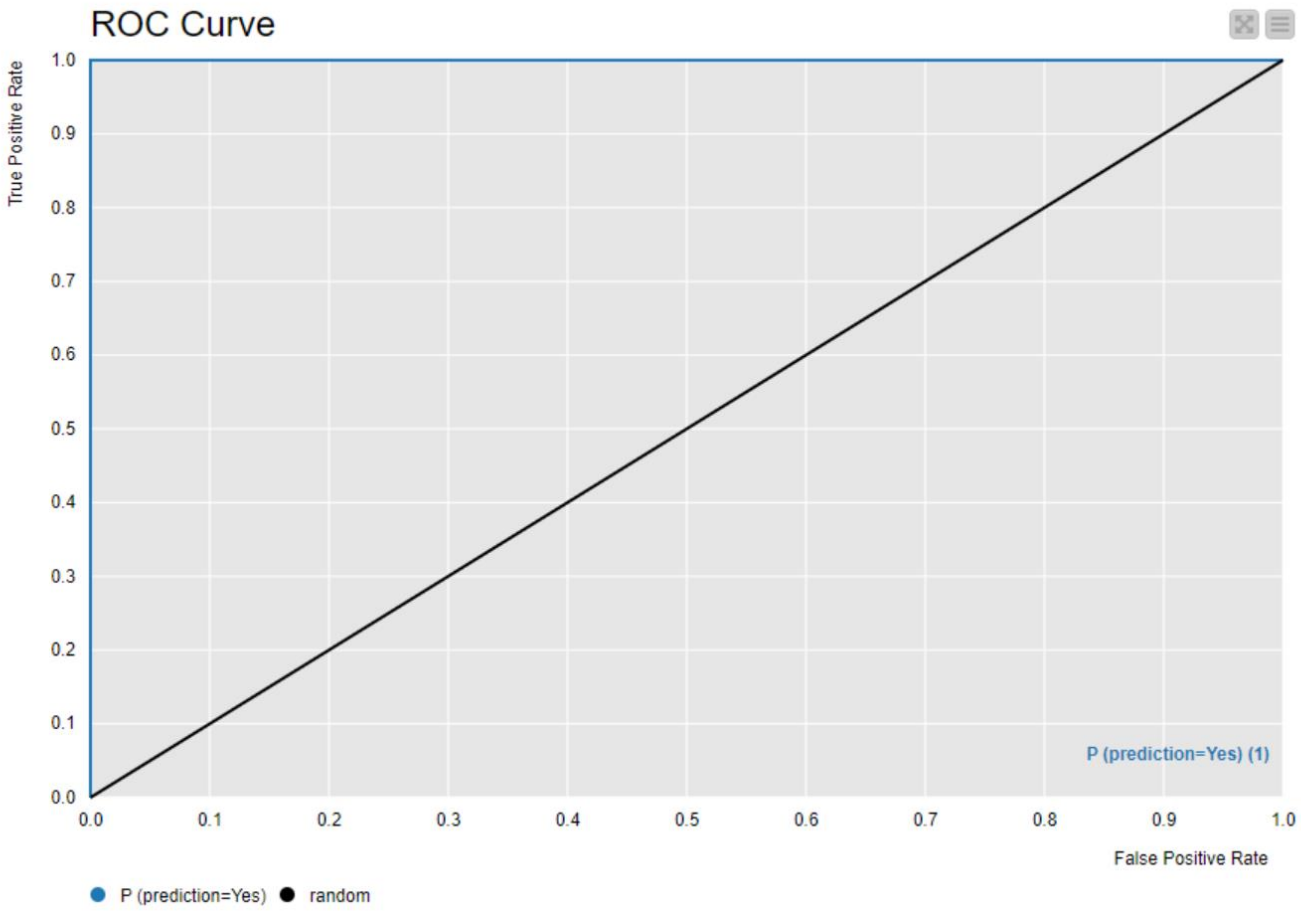
File Edit Hilite Navigation View

Table "spec_name" - Rows: 2 | Spec - Columns: 2 | Properties | Flow Variables

Row ID	No	Yes
No	6	90
Yes	5	165

Table "default" - Rows: 3 | Spec - Columns: 11 | Properties | Flow Variables

Row ID	TruePositives	FalsePositives	TrueNegatives	FalseNegatives	Recall	Precision	Sensitivity	Specificity	F-meas...	Accuracy	Cohen'...
No	6	5	165	90	0.062	0.545	0.062	0.971	0.112	?	?
Yes	165	90	6	5	0.971	0.647	0.971	0.062	0.776	?	?
Overall	?	?	?	?	?	?	?	?	?	0.643	0.041



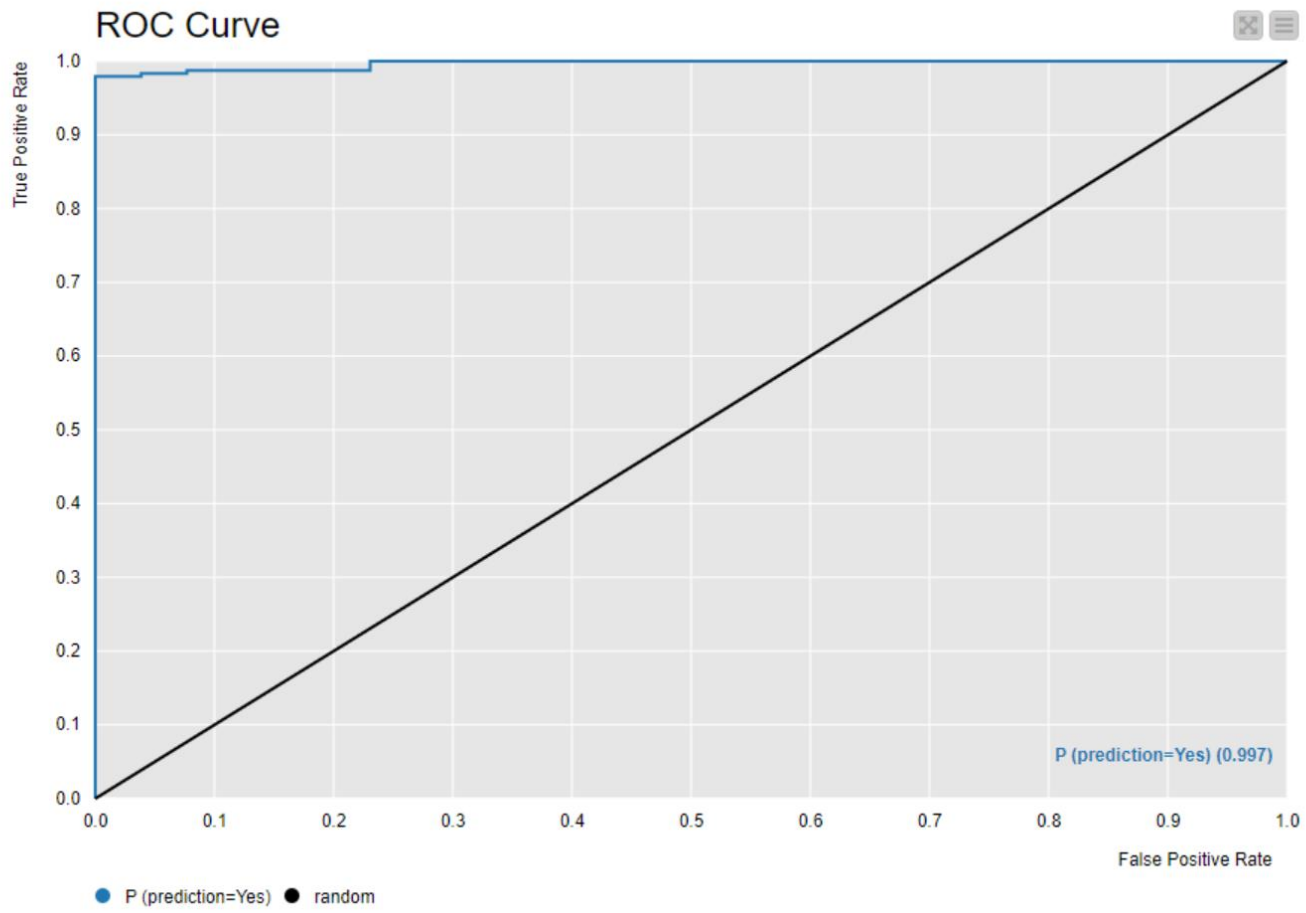
Appendix 4

Table "spec_name" - Rows: 2 | Spec - Columns: 2 | Properties | Flow Variables

Row ID	No	Yes
No	19	77
Yes	7	163

Table "default" - Rows: 3 | Spec - Columns: 11 | Properties | Flow Variables

Row ID	TruePositives	FalsePositives	TrueNegatives	FalseNegatives	Recall	Precision	Sensitivity	Specificity	F-meas...	Accuracy	Cohen'...
No	19	7	163	77	0.198	0.731	0.198	0.959	0.311	?	?
Yes	163	77	19	7	0.959	0.679	0.959	0.198	0.795	?	?
Overall	?	?	?	?	?	?	?	?	?	0.684	0.186



KNIME Workflow