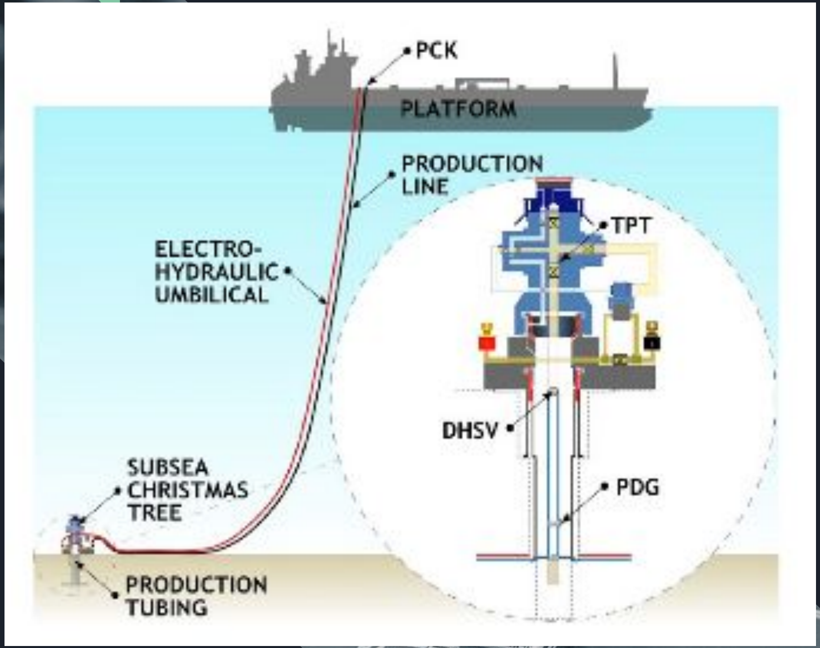
Strategic Thinking CA2

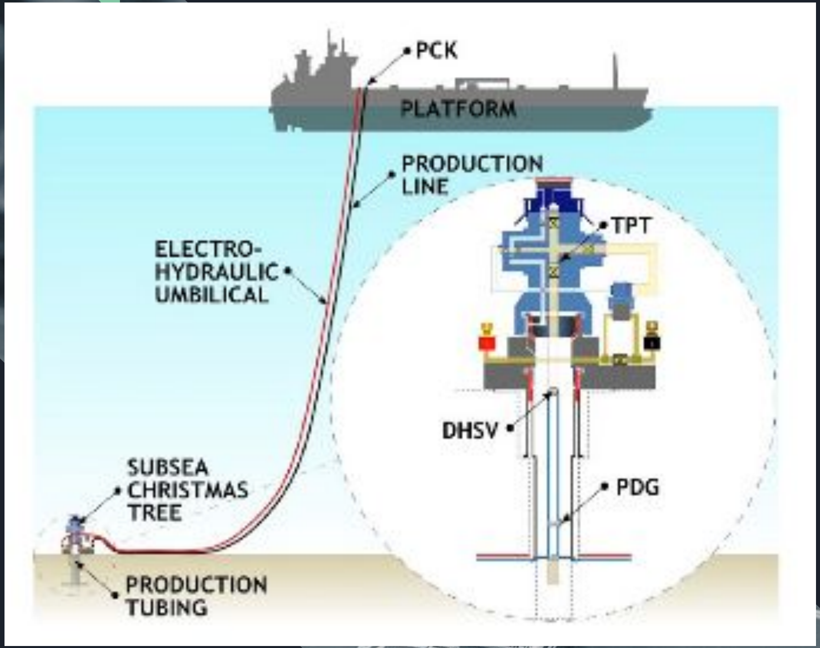
Giulio Calef, Kevin Byrne, Victor Ferreira Silva
HDip. in AI Applications - Sept. 2022

Business Description



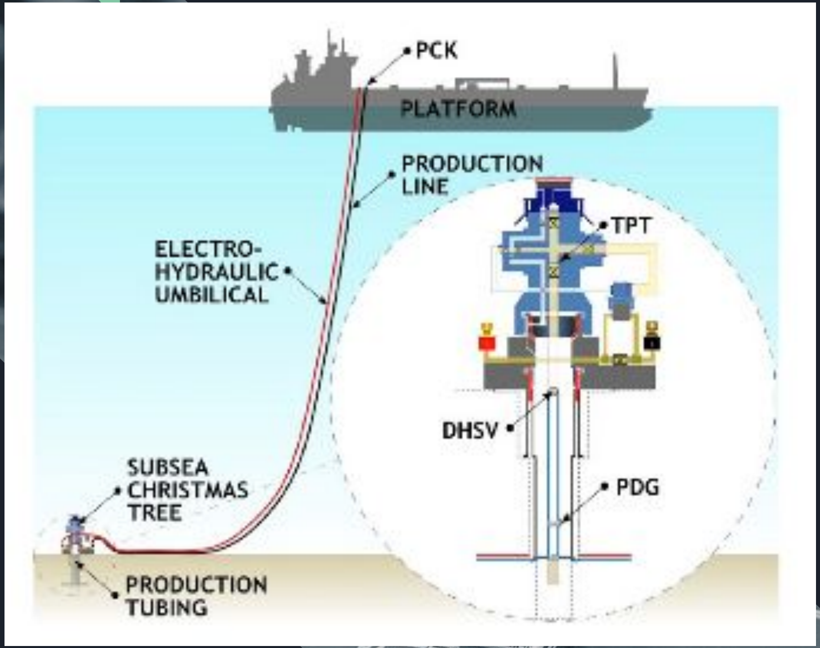
- Oil industry increasingly adopting automated controls for safer, more productive, and energy-efficient operations
- Timely detection of faults or anomalous systematic behaviors crucial to prevent production line disruptions
- Petrobras launched the "Expert Alarm Monitoring" project to improve abnormal event management in offshore wells

Business Description - Severe Slugging



- Severe Slugging is a critical flow assurance issue observed in offshore pipeline-riser systems
- Can cause flooding of downstream production facilities and decrease productivity
- Project contributes to mitigating safety risks, reducing operational costs, and improving production efficiency in offshore oil operations

Hypothesis & General Goal



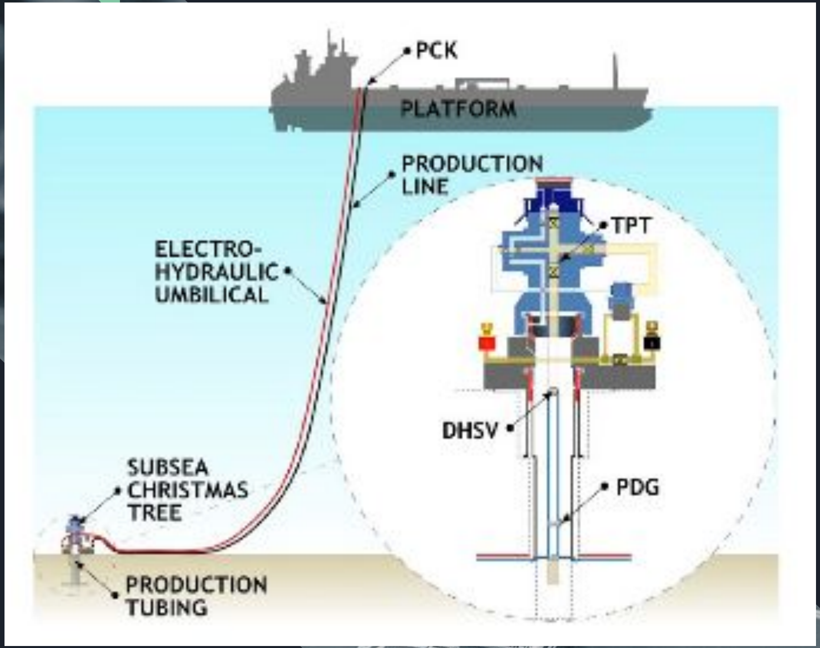
Hypothesis:

The data present in 3W Data Set's real instances enables classifier models to detect Severe Slugging with high accuracy, precision and recall.

General Goal:

Project objective: Apply ML to detect Severe Slugging in offshore well production using 3W dataset. Present a high-accuracy classification model.

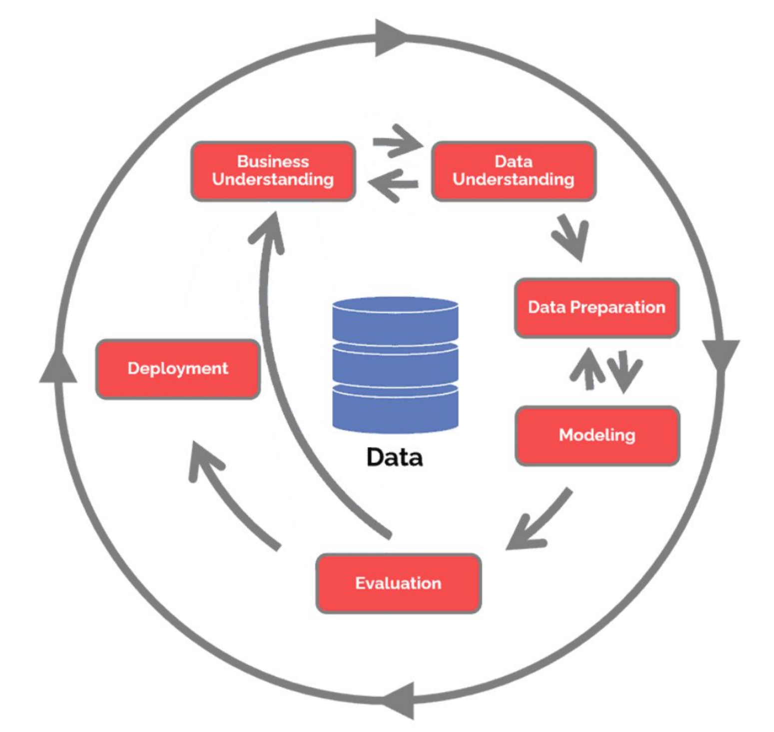
Success Criteria / Indicators



- Classification model to detect Severe Slugging in offshore wells using ML on the 3W data set.
- Metrics used are accuracy, precision, and recall.
- Adopting recall as a criterion helps evaluate minority class accuracy, and precision determines the probability of detecting Severe Slugging correctly.
- Using recall to consider class disparities in binary classification of imbalanced data.

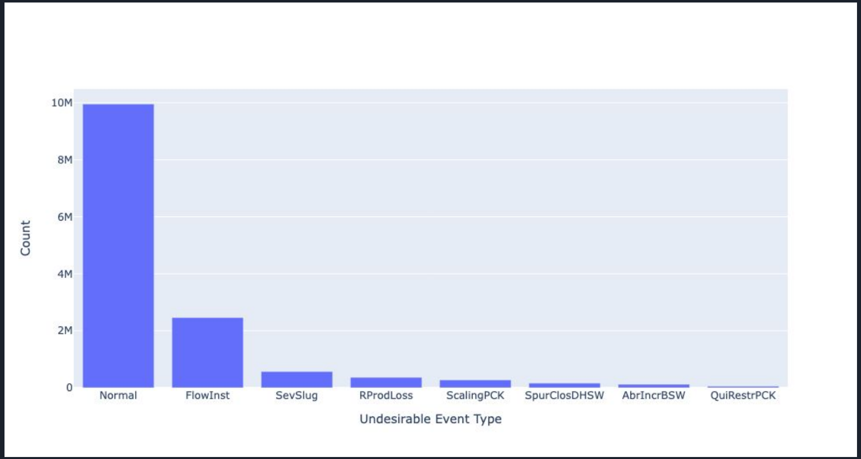
Technologies Used

- CRISP-DM methodology
- Petrobras' 3W Tool Kit
- Pandas, NumPy, Scikit-learn, Keras, Seaborn, Matplotlib, Plotly, Imbalanced-learn, and Pickle.
- Scikit-learn: LinearSVC, KNeighborsClassifier, DecisionTreeClassifier, and RandomForestClassifier
- modules: GridSearchCV, cross_val_score, train_test_split, KFold, and Pipeline.



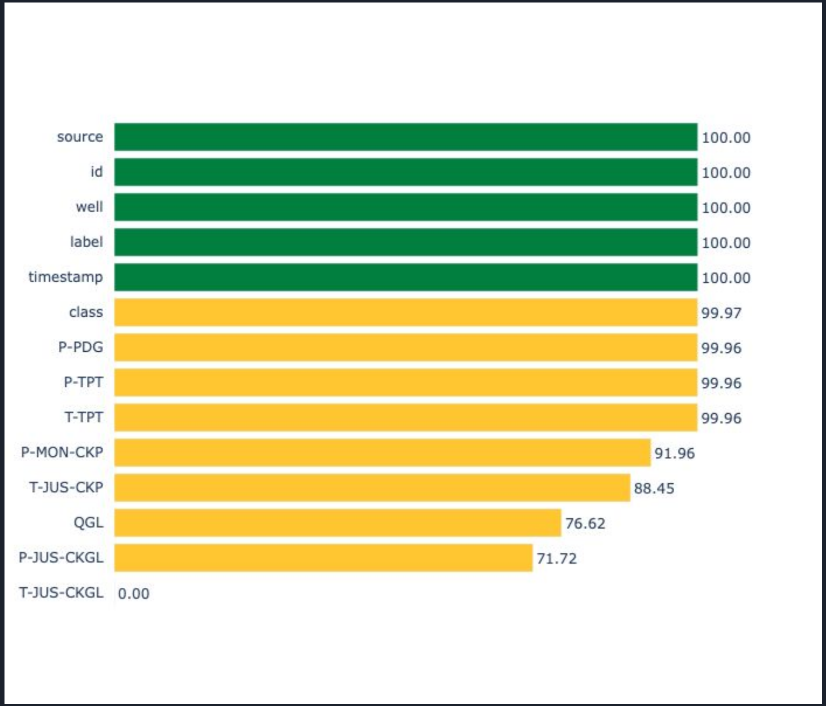
Data Understanding

- The dataset contains Real, Simulated, and Hand-drawn instances, but only Real instances were selected for this project.
- Data set includes 13,952,911 observations with 14 columns of data each x 8 types of undesirable events characterized by 8 process variables (real instances)
- Data set was not pre-processed, including NaN values, frozen variables, varying sizes, and outliers, to maintain realistic aspects.



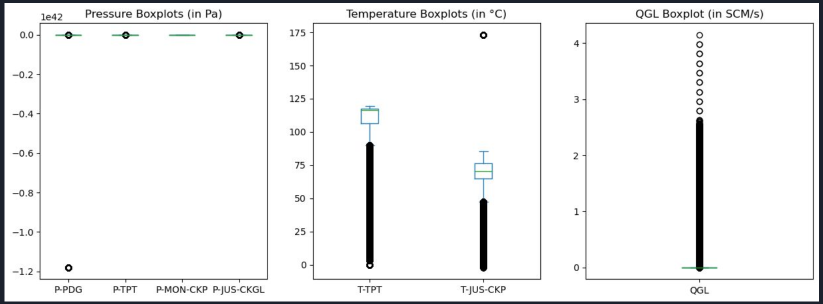
Data Understanding

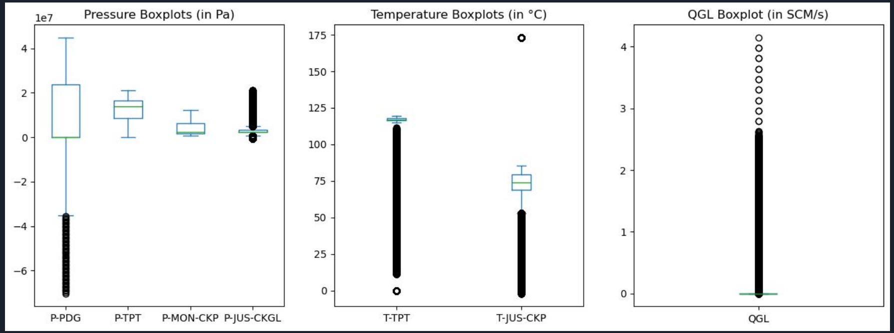
- Real, simulated, and hand-drawn instances were present in 3W data set, but only real instances were selected for this project
- Data set includes 13,952,911 observations with 14 columns of data each x 8 types of undesirable events characterized by 8 process variables (real instances)
- Data set was not pre-processed, including NaN values, frozen variables, varying sizes, and outliers, to maintain realistic aspects.



Data Preparation

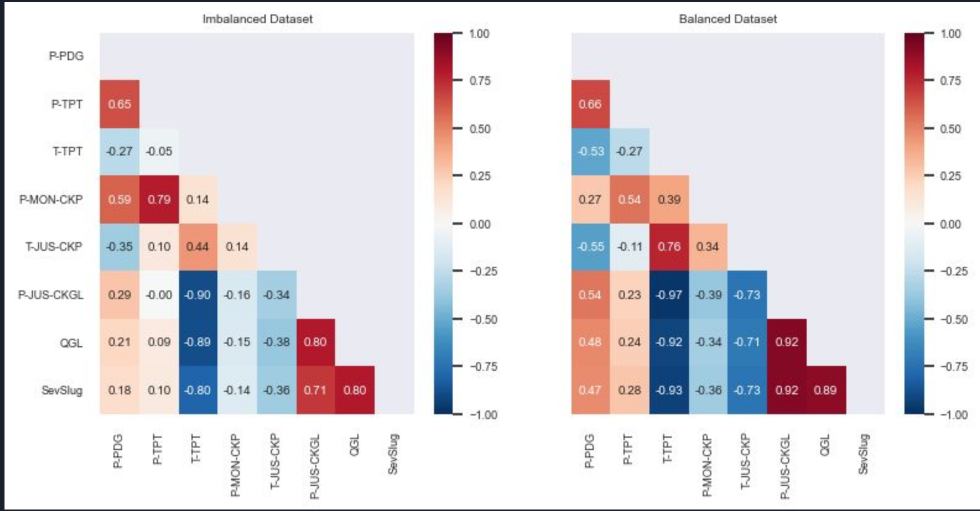
- Data cleaning involved removing missing data from certain columns, dropping redundant columns, and removing duplicates.
- Extreme outliers in P-PDG and P-TPT were also removed, which represented 2.26% of the resulting rows.
- The resulting distribution of values in P-PDG and P-TPT was modified.





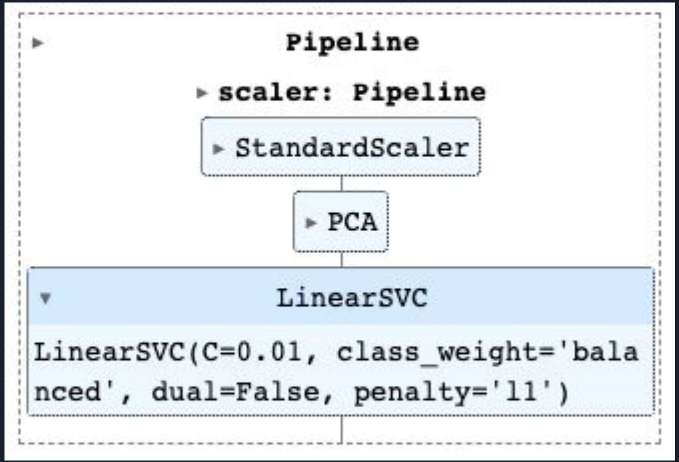
Feature Engineering

- Boolean columns were created for each undesirable event, and the dataset was split into training and testing sets.
- The distribution of records with or without severe slugging was also computed
- Non-Severe Slugging: 94.194%
- Severe Slugging: 5.806%



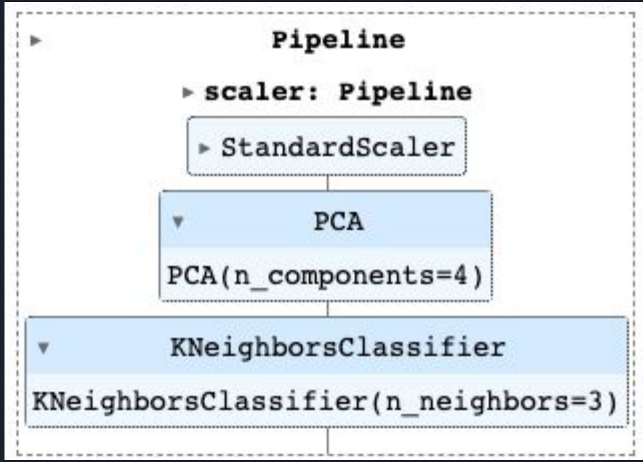
Models - LinearSVC

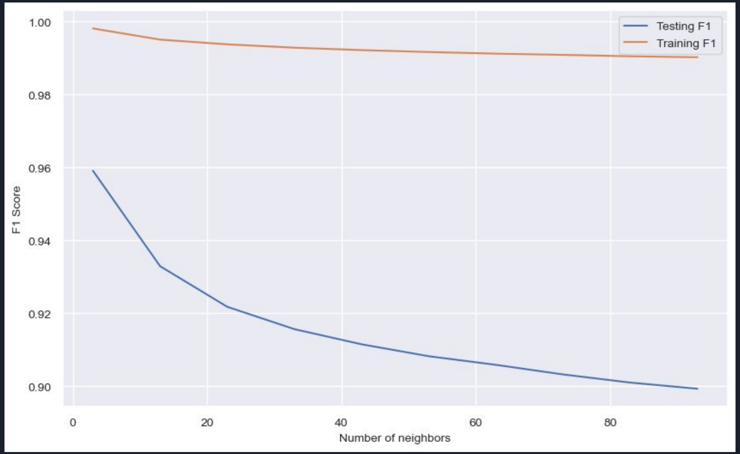
- LinearSVC: a linear support vector classifier used for binary classification.
- Data was not linearly separable
- Hyperparameter optimization + pipeline with StandardScaler, PCA, and model.
- The best parameters found: StandardScaler, PCA with 3 components, and a LinearSVC with C=0.01, class weight='balanced', dual=False, and penalty='l1'.



Models - KNN

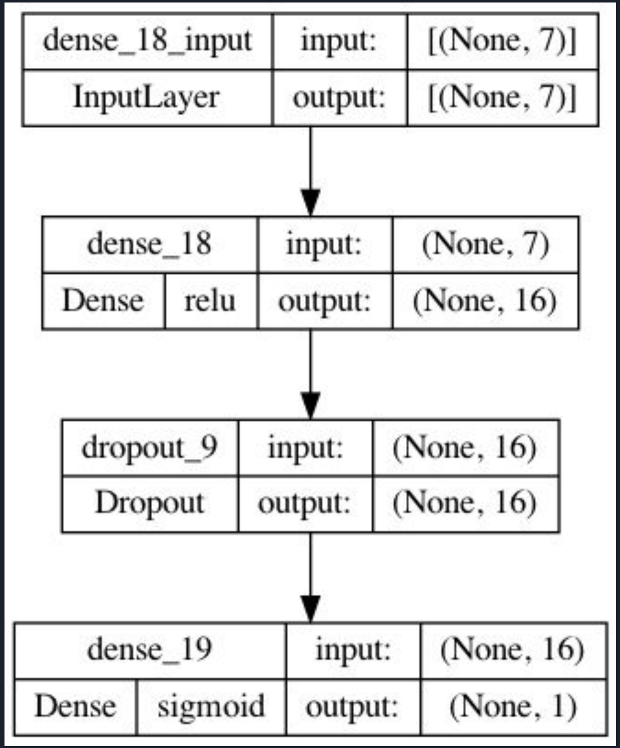
- Pipeline with 3 steps: StandardScaler method to scale the data, PCA and model.
- Grid search + default cross-validation (5-fold cross-validation)
- Optimal combination of parameters found
- Comparison between the f1 score in training and test data sets according to the number of neighbors





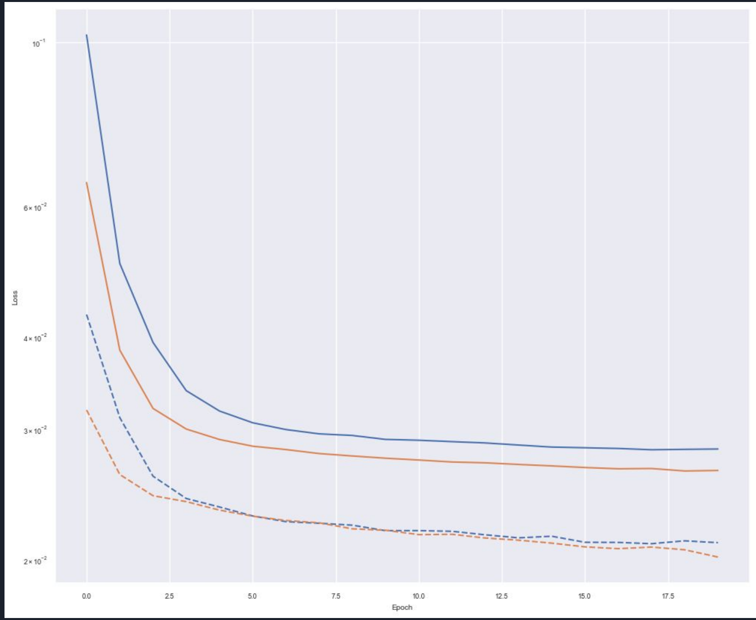
Models - Neural Networks

- The peril of overfitting in an imbalanced dataset: data was split into train and validation sets,
- Validation set only used for evaluation during training
- The test set was completely isolated until evaluation.
- Lack of training data for the Severe Slugging class.
- Data cleaning and scaling followed the same process as other models in the project.



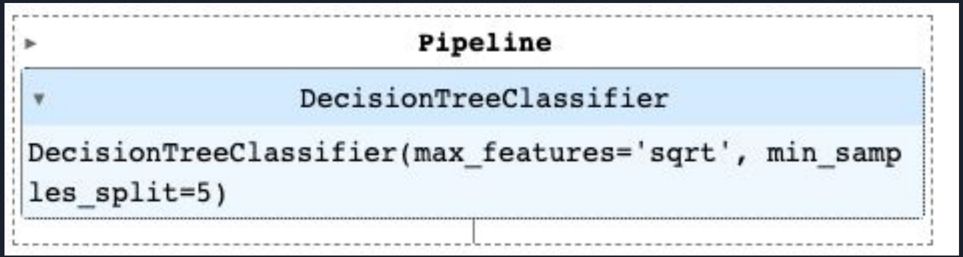
Models - Neural Networks

- An initial model was defined and bias was fixed by adjusting the loss for imbalanced data.
- Trained with 20 epochs and compared to the same model without the bias adjustment, showing that loss was significantly reduced.
- Training history was recorded to verify over-fitting and to collect data for visualising relevant metrics.
- The model was trained with 100 epochs while the precision-recall curve of validation data set was being monitored - lastly 58 epochs

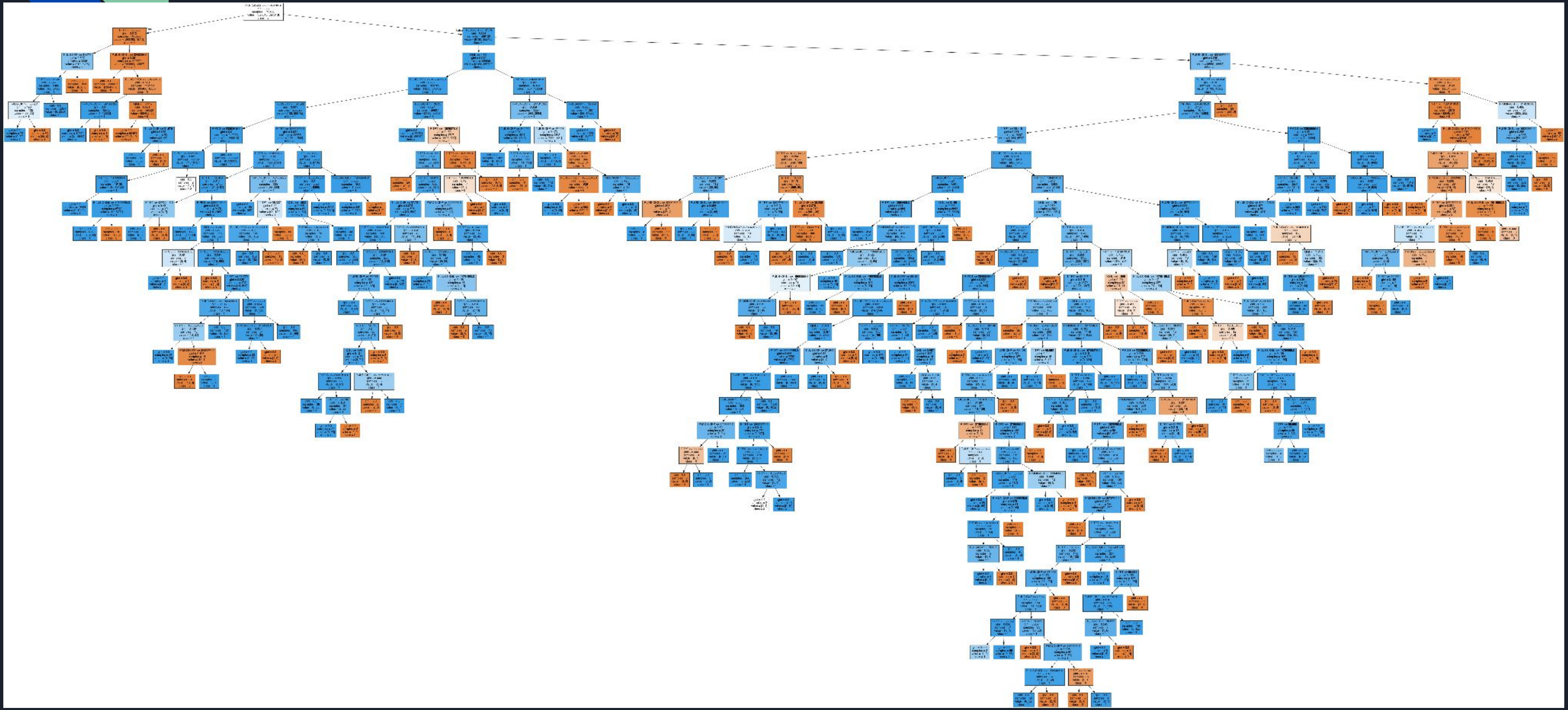


Models - Decision Tree

- Decision Tree Classifier is used as a non-linear classifier for non-linearly separable data.
- Hyperparameters were optimized using GridSearchCV and the best combination was used to train the model.
- The resulting decision tree was too complex to be visualized in detail.

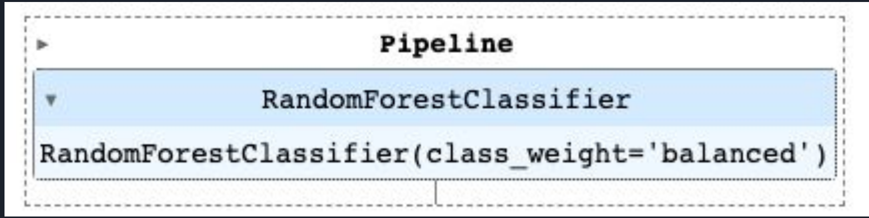


Models - Decision Tree



Models - Random Forest

- A Random Forest Classifier was used to classify non-linear data.
- A previously trained model was loaded to predict class labels for X test and generate a classification report.
- Hyperparameter optimization was performed using a grid search and the best parameters were used to train the model.
- The final pipeline used RandomForestClassifier with class_weight set to 'balanced'.



Evaluation - Classification Reports

Linear SVC

```
# printing classification report for LinearSVC
print(cr_linearsvc)
```

```
              precision    recall  f1-score   support

           0     0.9980    0.9808    0.9893   2763432
           1     0.7568    0.9687    0.8498    170839

    accuracy                         0.9801   2934271
   macro avg     0.8774    0.9747    0.9195   2934271
weighted avg     0.9840    0.9801    0.9812   2934271
```

- Lowest precision, accuracy

Evaluation - Classification Reports

k-Neighbours Classifier

```
# printing classification report for kNN classifier
print(cr_knn)
```

```
              precision    recall  f1-score   support

           0    0.99987   0.99487   0.99736   2763432
           1    0.92328   0.99787   0.95913    170839

    accuracy                        0.99505   2934271
   macro avg    0.96157   0.99637   0.97825   2934271
weighted avg    0.99541   0.99505   0.99514   2934271
```

- Satisfactory accuracy, recall

Evaluation - Classification Reports

Neural Networks

```
# printing classification report for ANN
print(cr_ann)
```

```
              precision    recall  f1-score   support

           0    0.99321   0.99987   0.99653   2763432
           1    0.99760   0.88938   0.94039    170839

    accuracy                        0.99343   2934271
   macro avg    0.99541   0.94462   0.96846   2934271
weighted avg    0.99346   0.99343   0.99326   2934271
```

- Satisfactory f1 score, but not good recall

Evaluation - Classification Reports

Decision Tree

```
# printing classification report for Decision Tree
print(cr_tree)
```

	precision	recall	f1-score	support
0	0.99999	0.99972	0.99986	2763432
1	0.99553	0.99980	0.99766	170839
accuracy			0.99973	2934271
macro avg	0.99776	0.99976	0.99876	2934271
weighted avg	0.99973	0.99973	0.99973	2934271

- Very high precision, recall, accuracy and f1-score

Evaluation - Classification Reports

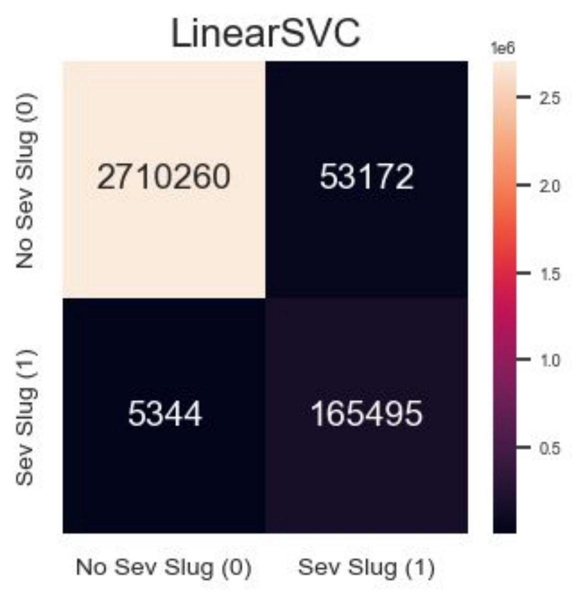
Random Forest

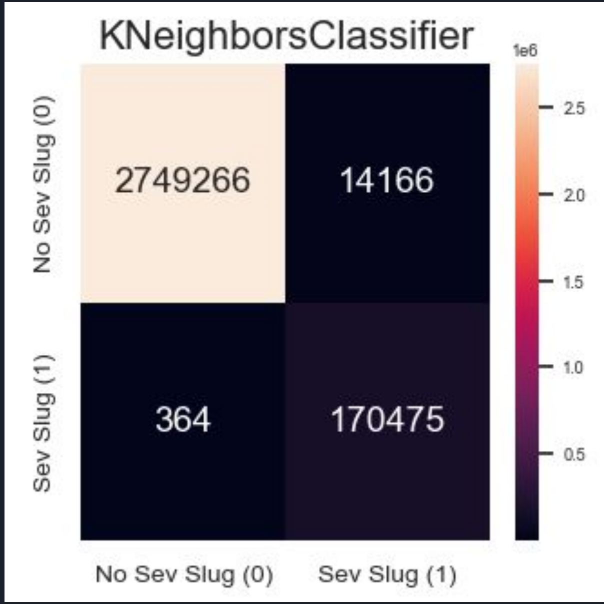
```python
# printing classification report for Random Forest
print(cr_rf)
```

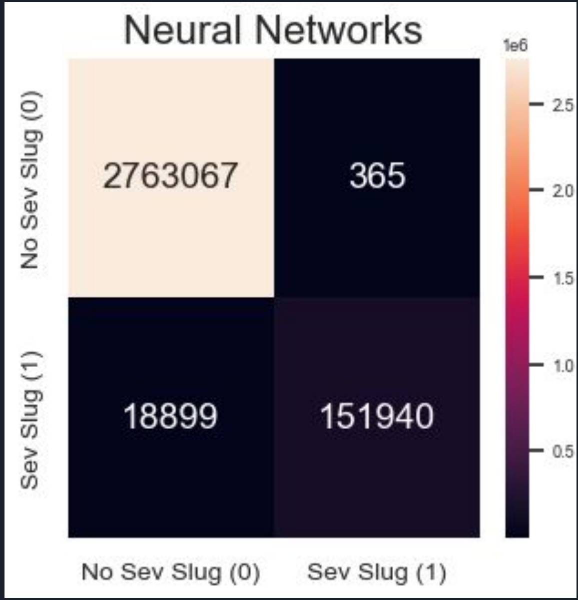
	precision	recall	f1-score	support
0	1.00000	0.99991	0.99995	2763432
1	0.99852	1.00000	0.99926	170839
accuracy			0.99991	2934271
macro avg	0.99926	0.99995	0.99961	2934271
weighted avg	0.99991	0.99991	0.99991	2934271

- Very high precision, recall, accuracy and f1-score

Evaluation - Confusion Matrices

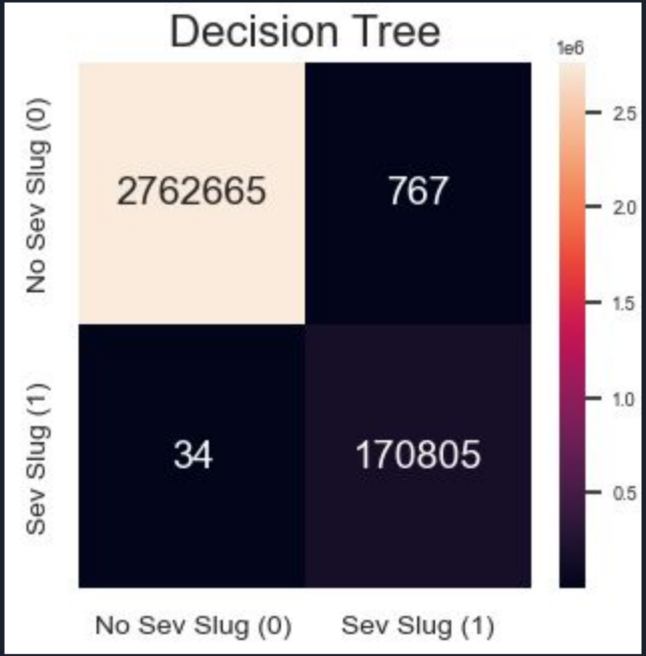


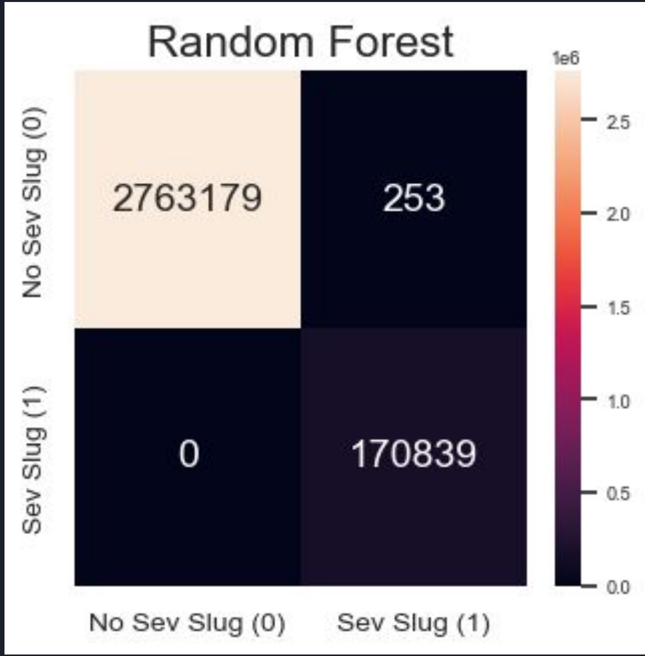




- Neural Networks had few False Negatives, but a very high False Positives

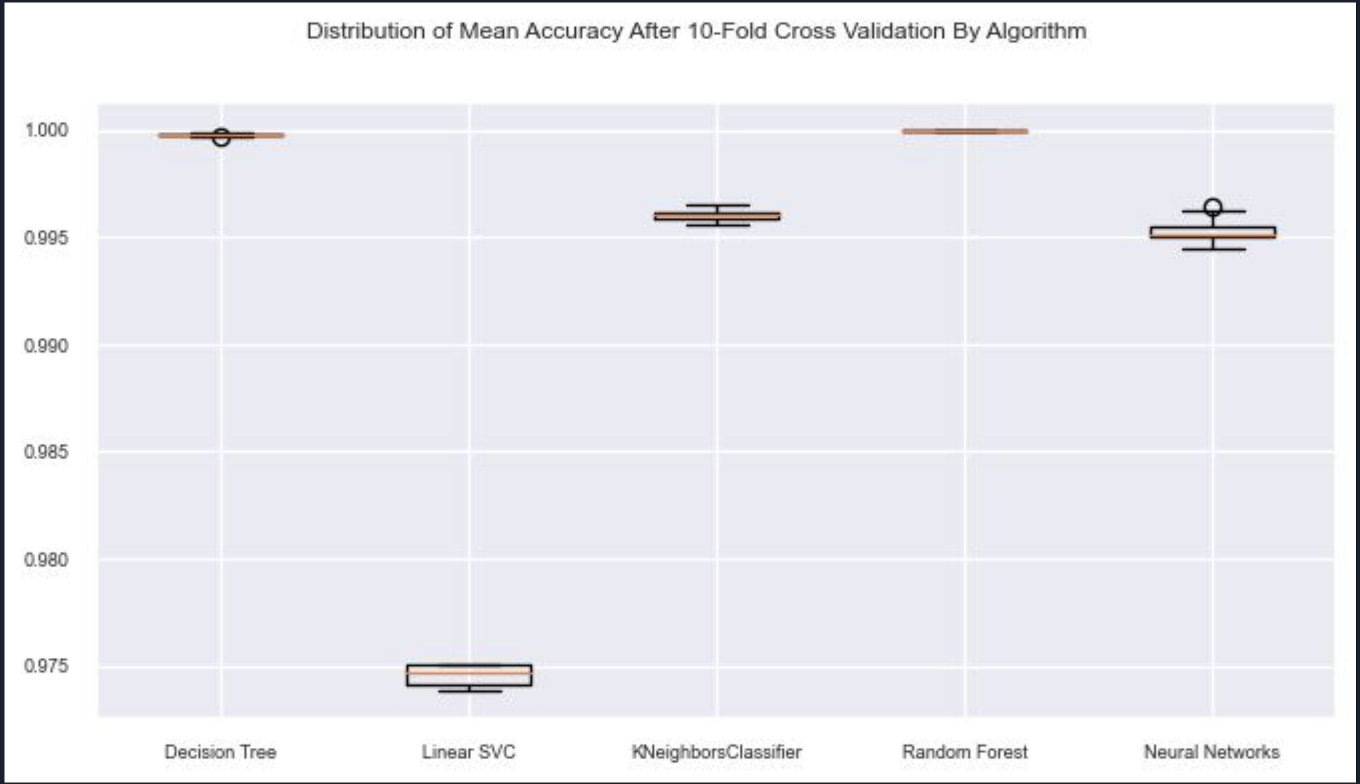
Evaluation - Confusion Matrices



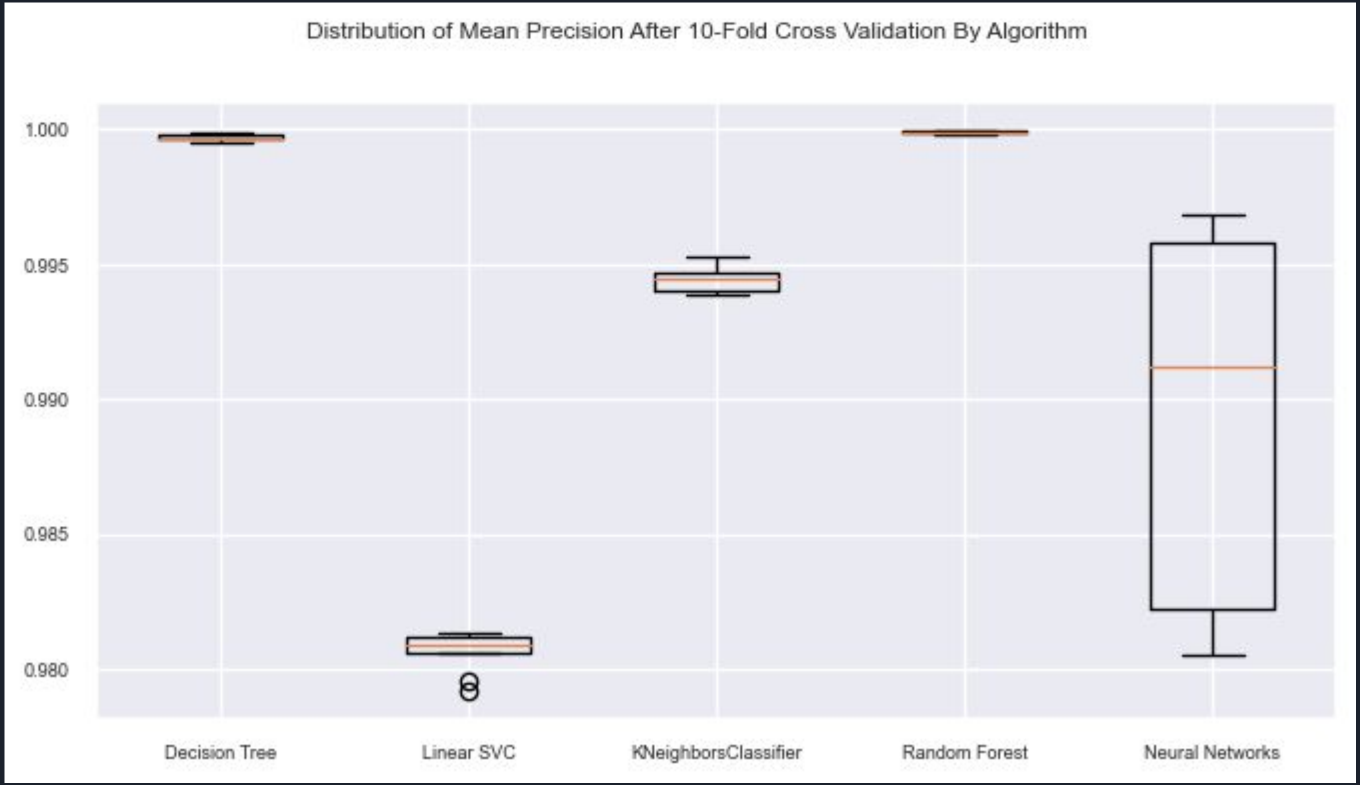


- Lowest numbers of false negatives and false positives

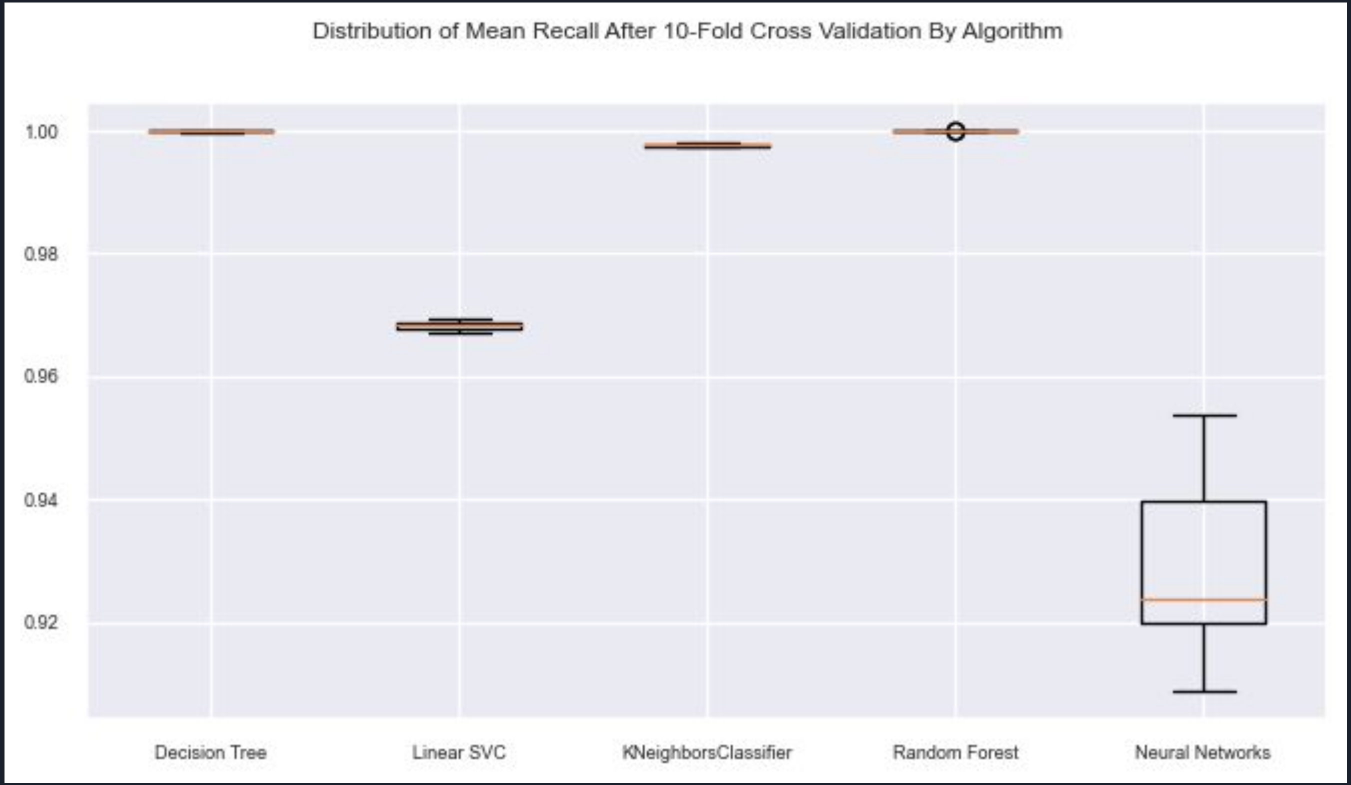
Evaluation - Accuracy



Evaluation - Precision



Evaluation - Recall



Conclusions

- Capstone project developed ML models to detect Severe Slugging in offshore well production lines
- Random Forest and Decision Tree classifiers showed very satisfactory results in all selected metrics
- The models can reduce operational and environmental risks, costs, and improve production efficiency.
- Techniques used can be applied to detect other undesirable events in the oil and gas industry