



•

BeCode

PROPUESTA TITANIC

MISAELE
VICTOR
HECTOR
GERMAN

LINKS

Link de coolab

https://colab.research.google.com/github/Misaelchavez16/Titanic_Machine_Learning_from_Disaster/blob/main/Titanic.ipynb

Link de github

https://github.com/Misaelchavez16/Titanic_Machine_Learning_from_Disaster

CONTEXTO



Análisis del proyecto

El reto trata de analizar los datos de los pasajeros que se encontraban en el Titanic y, con estos, realizar feature engineering, limpieza, entre otras cosas, para poder entrenar un modelo de machine learning y así predecir la supervivencia de los pasajeros dependiendo de varios feutures."



ELEMENTOS UTILIZADOS



Scikit Learn

librería gratuita de ciencia de datos y consultoría de inteligencia artificial. Ofrece una amplia gama de herramientas, como algoritmos para imputar datos, detección de anomalías, codificación de etiquetas, reducción de dimensionalidad y clasificación.



Numpy

Biblioteca especializada en cálculo numérico y análisis de datos. Introduce sus propios arreglos, hasta 50 veces más eficientes en términos de velocidad que las listas estándar de Python.



Seaborn

Es una biblioteca enfocada en la creación de gráficos estadísticos. Se basa en Matplotlib pero aporta simplicidad y funciones inéditas.



Pandas

Pandas funciona como una herramienta para carga, manipulación y fusión de datos, de manera eficiente; siendo esta una de las más populares.

EXPLORACIÓN Y PREPROCESAMIENTO DE LOS DATOS

DATOS FALTANTES

01

Verificar los datos faltantes

Utilizando la libreria de missingno, utilizamos la funcion de matrix para observar los datos faltantes en cada atributo del dataset de una manera más visual y clara.

- Los valores n umericos iguales o menores a zero, asignaron con “np.nan”.

02

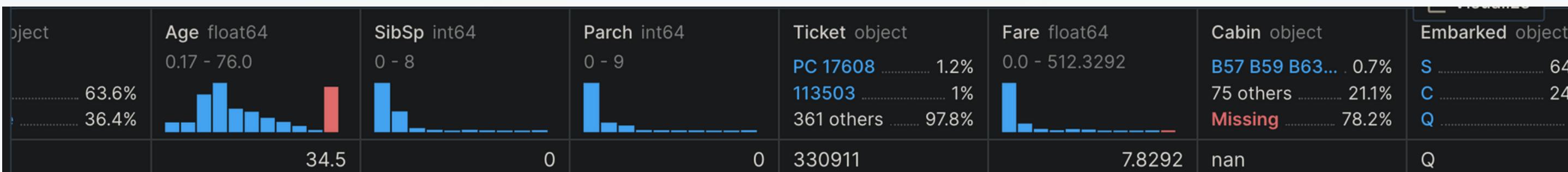
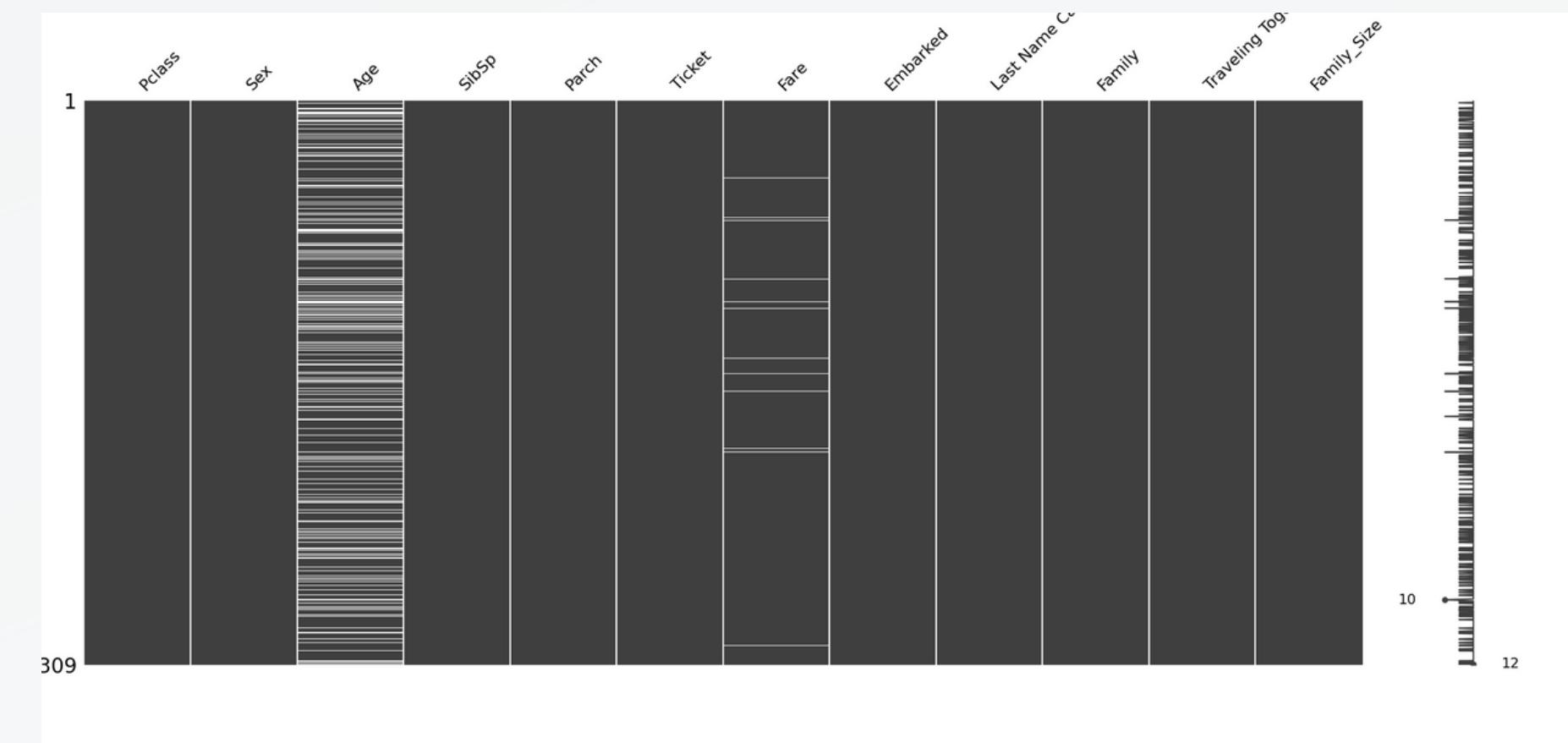
Aplicar Knn

Utilizando el algoritmo de “knn imputer”, llenamos los datos faltantes que se encontraron en el paso anterior.

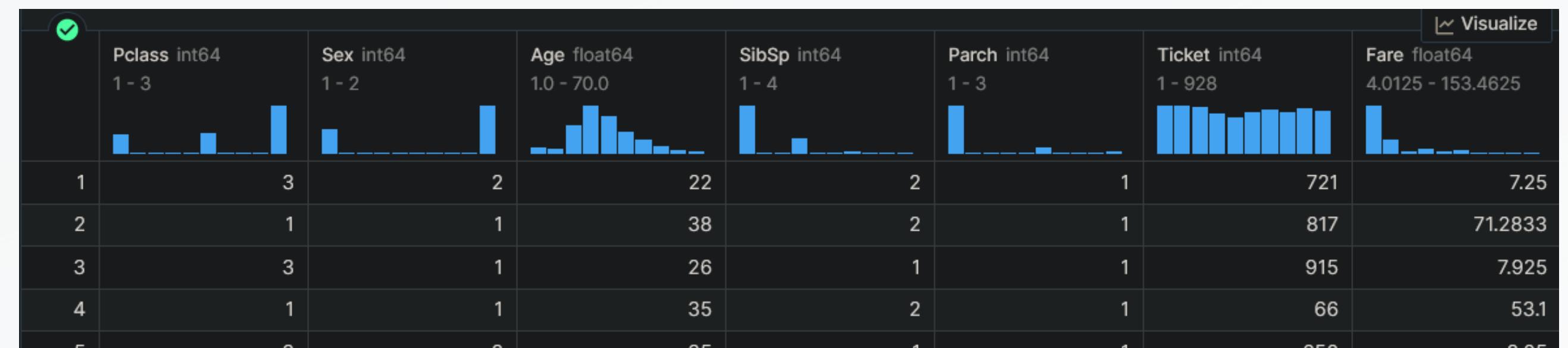
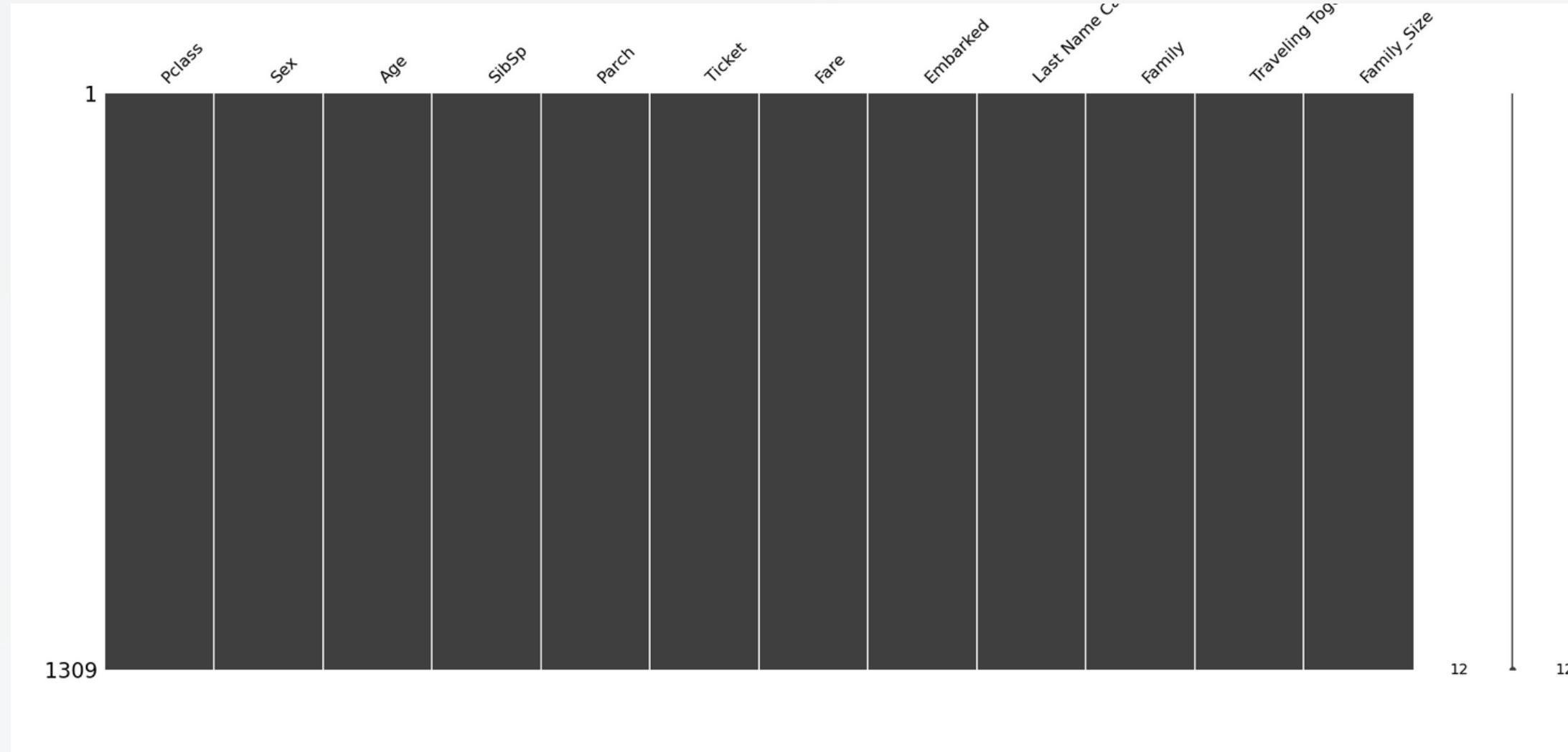
03

Eliminaci n de datos

Existian atributos con grandes cantidades de datos faltantes, por lo que decidimos eliminarlos.



DATASET COMPLETO

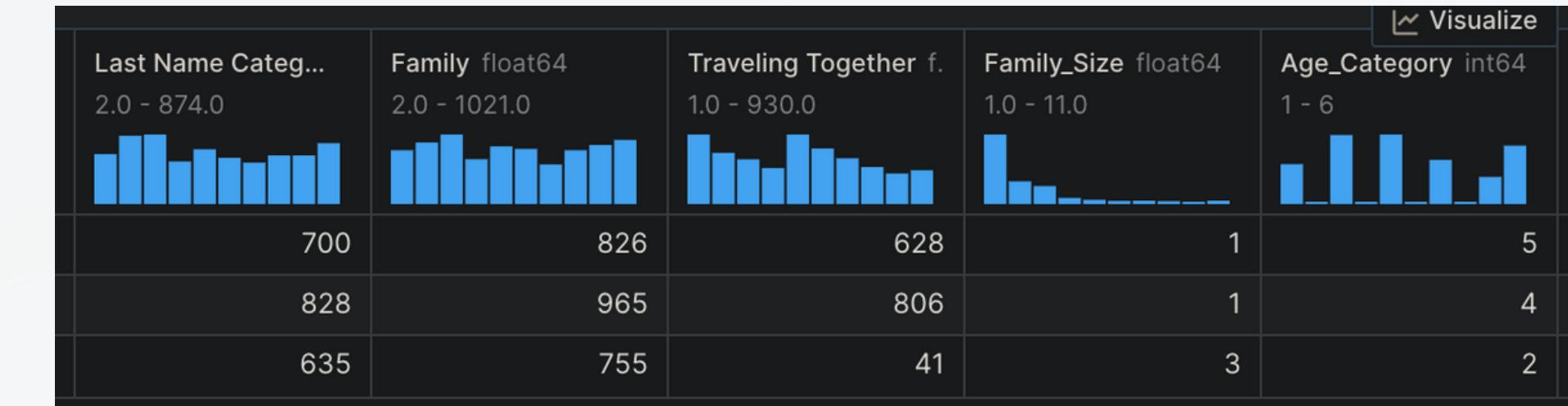


TRANSFORMACION DE DATOS

01

Agregar nuevas columnas

Agregamos nuevas columnas para aumentar el número de atributos, se agregaron Last Name, Traveling Together y Family.

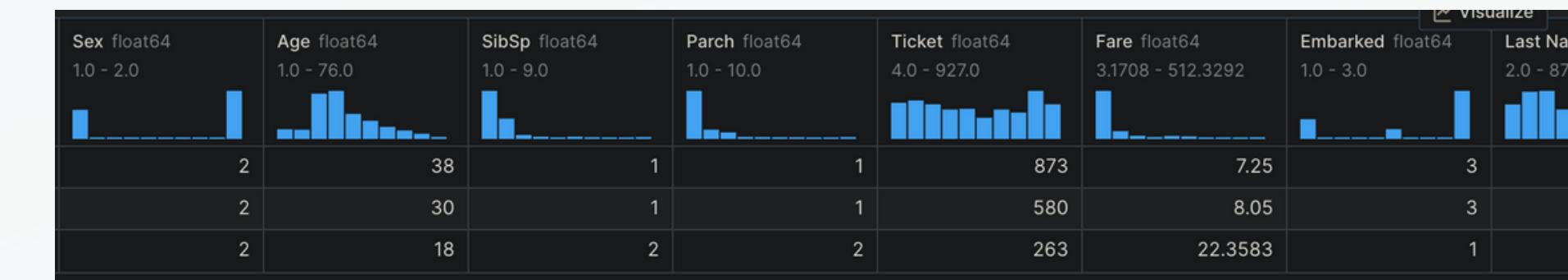


02

Identificación de datos

Identificamos los datos categóricos que tenemos y que podemos convertirlos a numéricos para mejor tratamiento de outliers.

- Se aplico “label encoding”, para mejor tratamiento de outliers y valores faltantes.



03

Detección de outliers

- Se calculó los puntajes z-scores para una columna específica en un DataFrame. Los valores atípicos se identifican si su puntaje Z excede el umbral definido.
- Se utilizó el algoritmo de Isolation Forest. Este funciona mediante la creación de árboles de decisión que aíslan los valores atípicos del resto de los datos, lo que facilita su identificación y tratamiento.
- Se eliminan las muestras atípicas.

DISTRIBUCION Y ANALISIS

01

Correlation Matrix

La matriz de correlación revela relaciones entre diversas características en el conjunto de datos, como la correlación negativa entre "Clase" y "Tarifa," la correlación negativa entre "Sexo" y "Sobrevivientes," y la correlación positiva entre "SibSp" y "Tamaño de la Familia," lo que sugiere conexiones importantes en los datos.

02

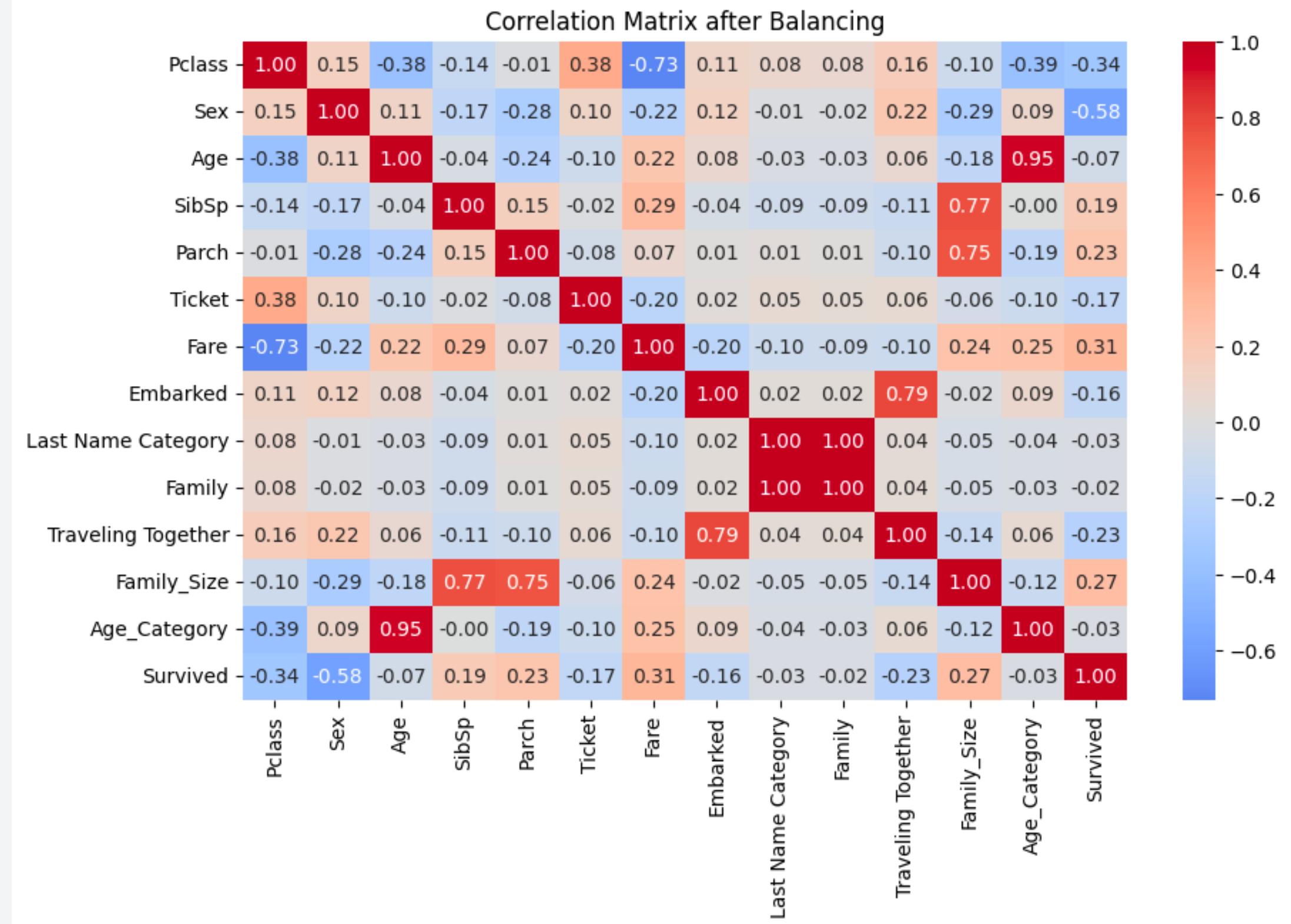
Histogramas con curva de densidad

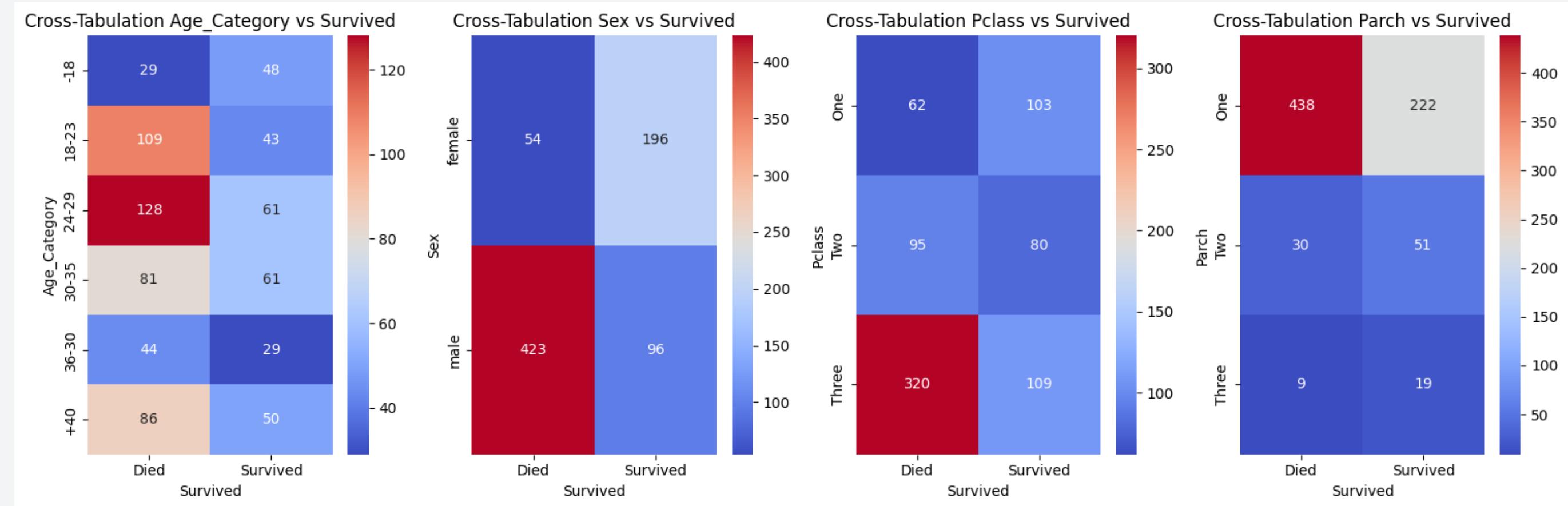
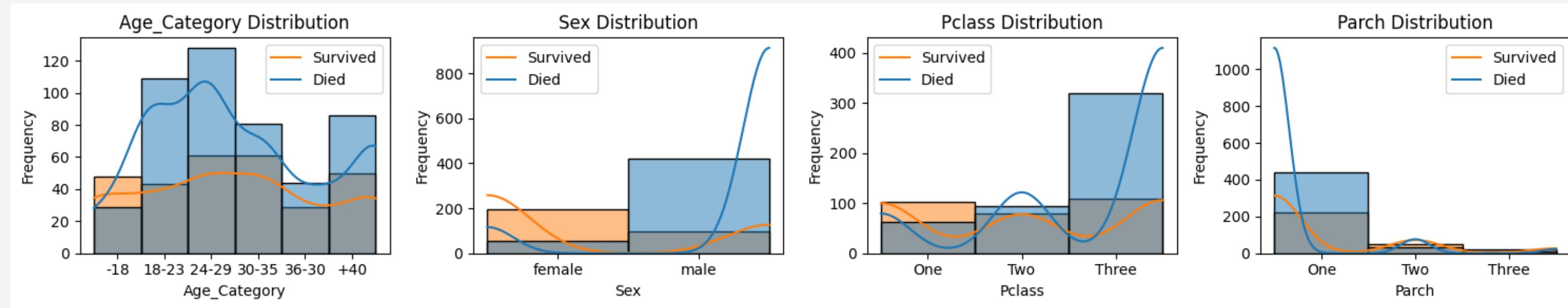
Utilizamos gráficas de distribución para observar la distribución de los datos con respecto a los sobrevivientes y los fallecidos

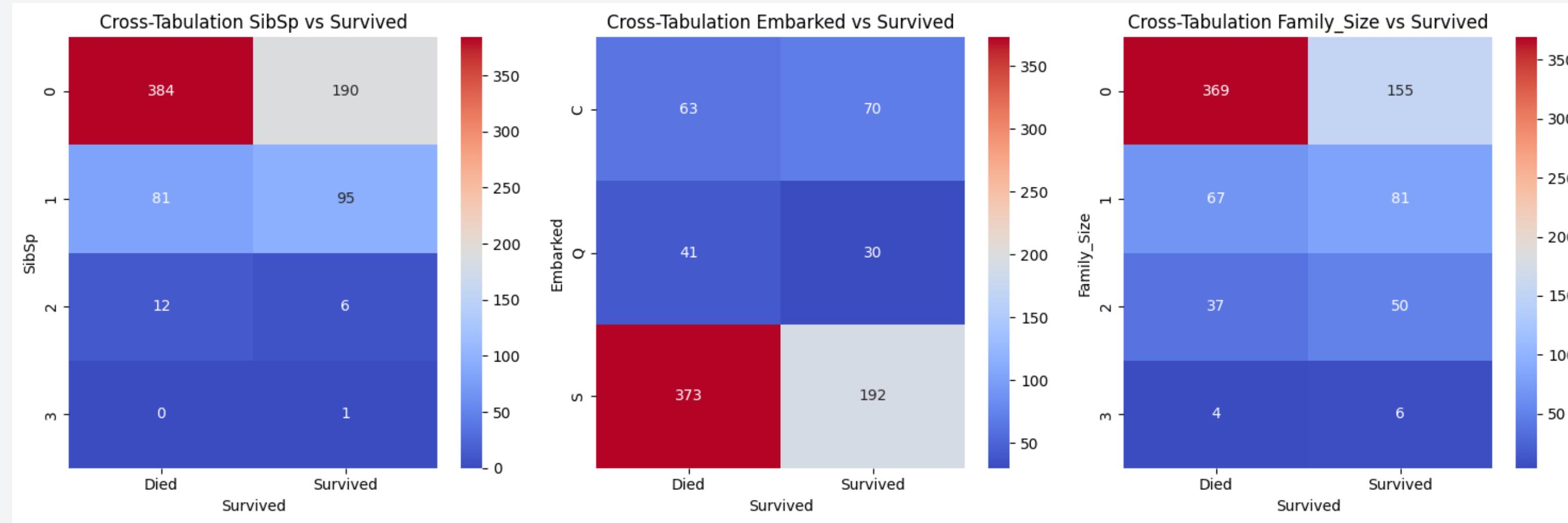
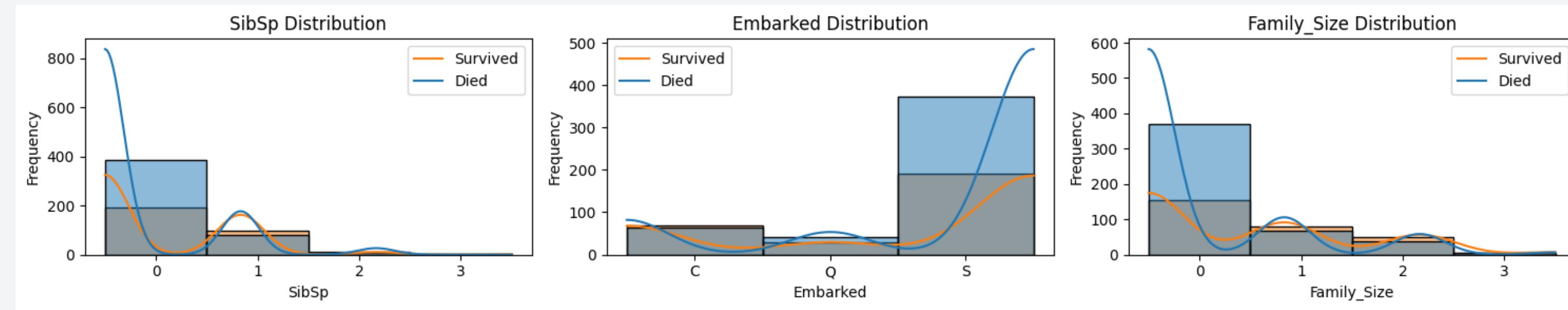
03

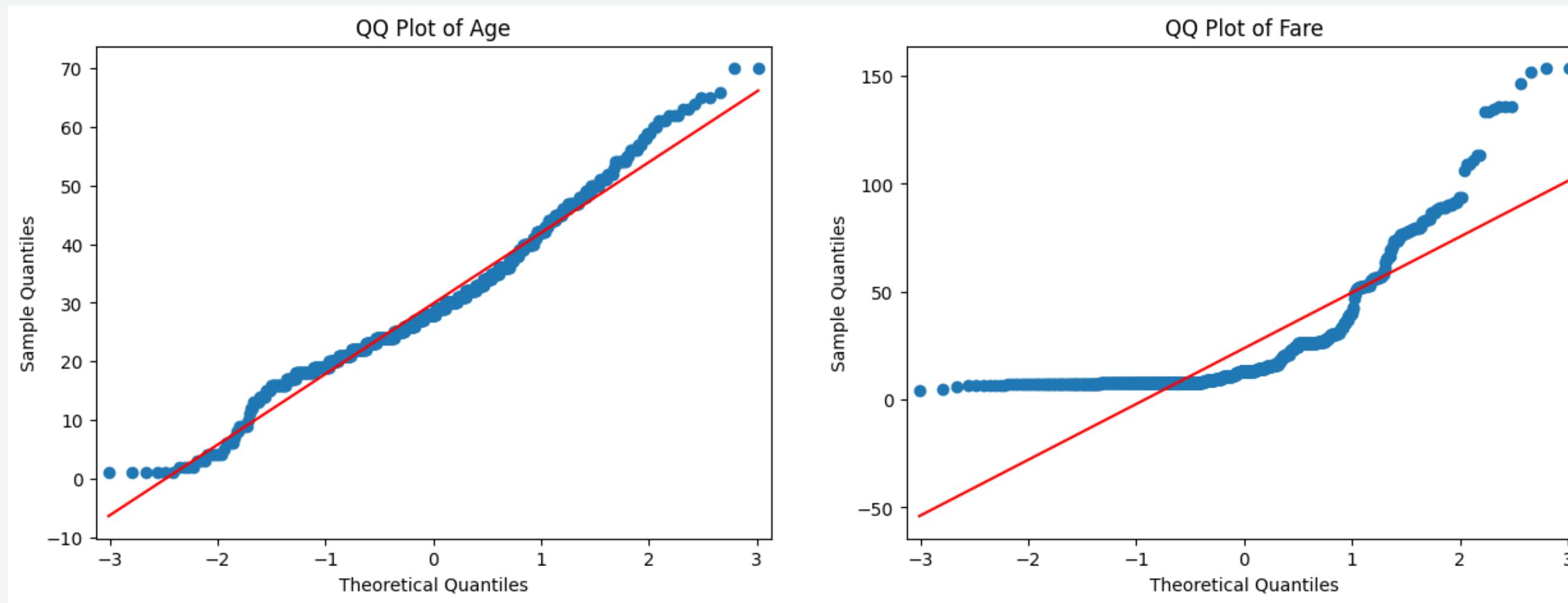
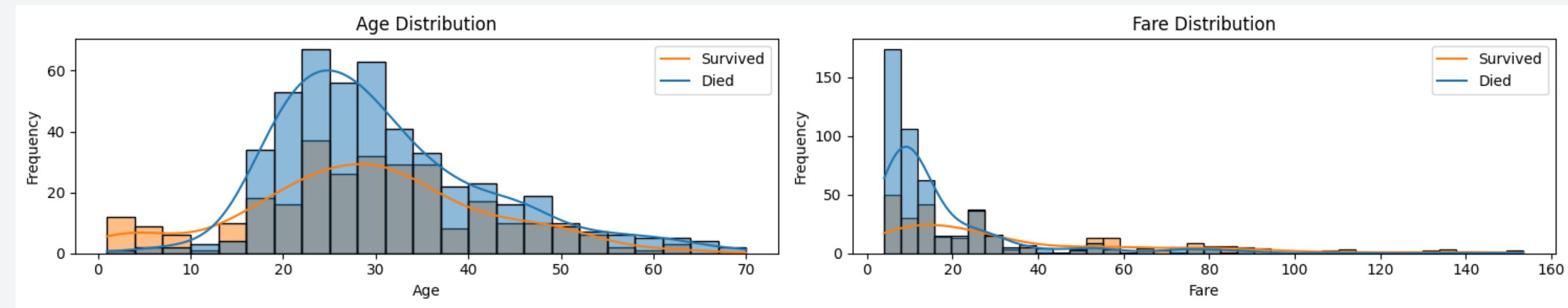
Cross tabulation plots

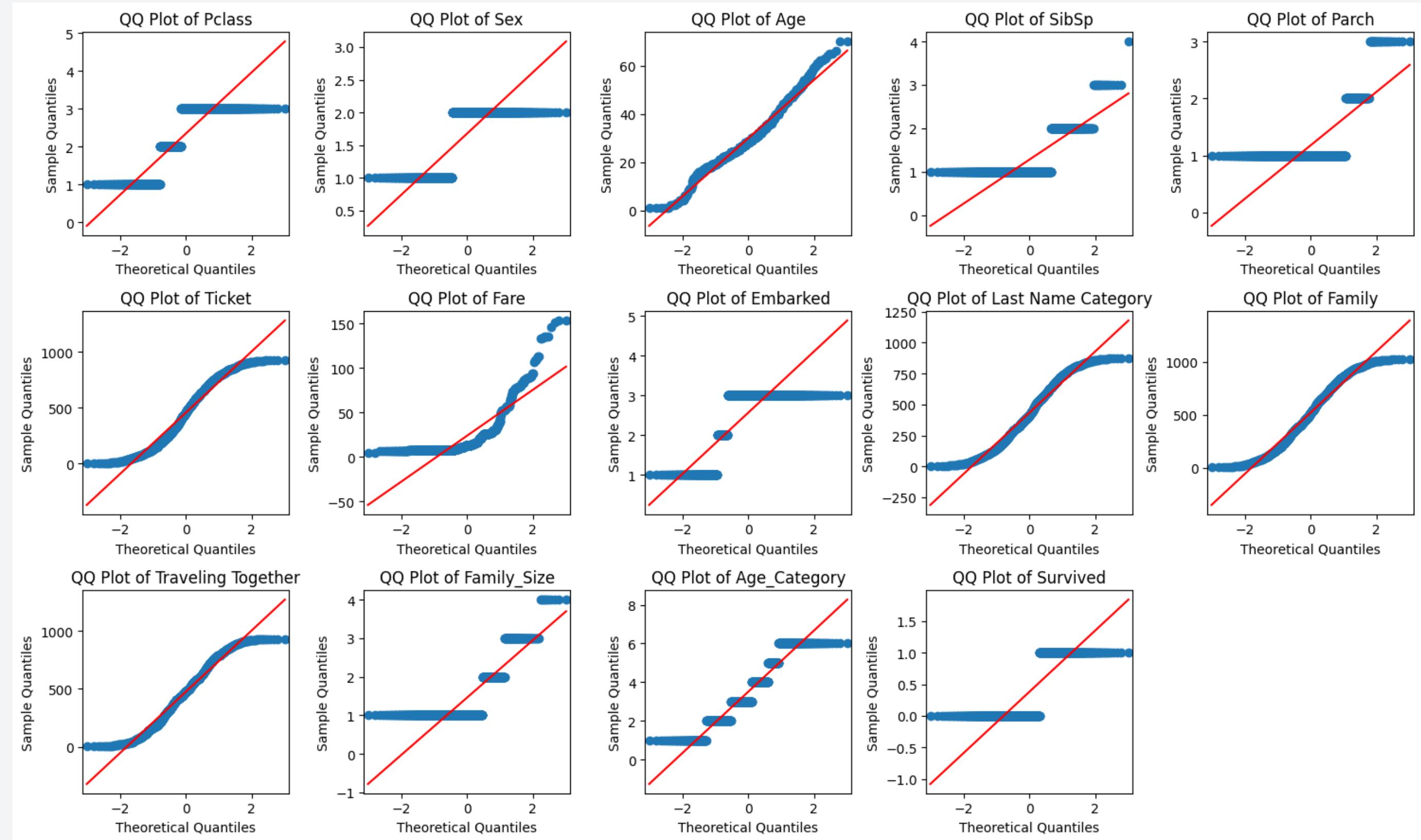
- Histogramas y líneas de densidad revelan la distribución y la tendencia central de los datos de manera visual.

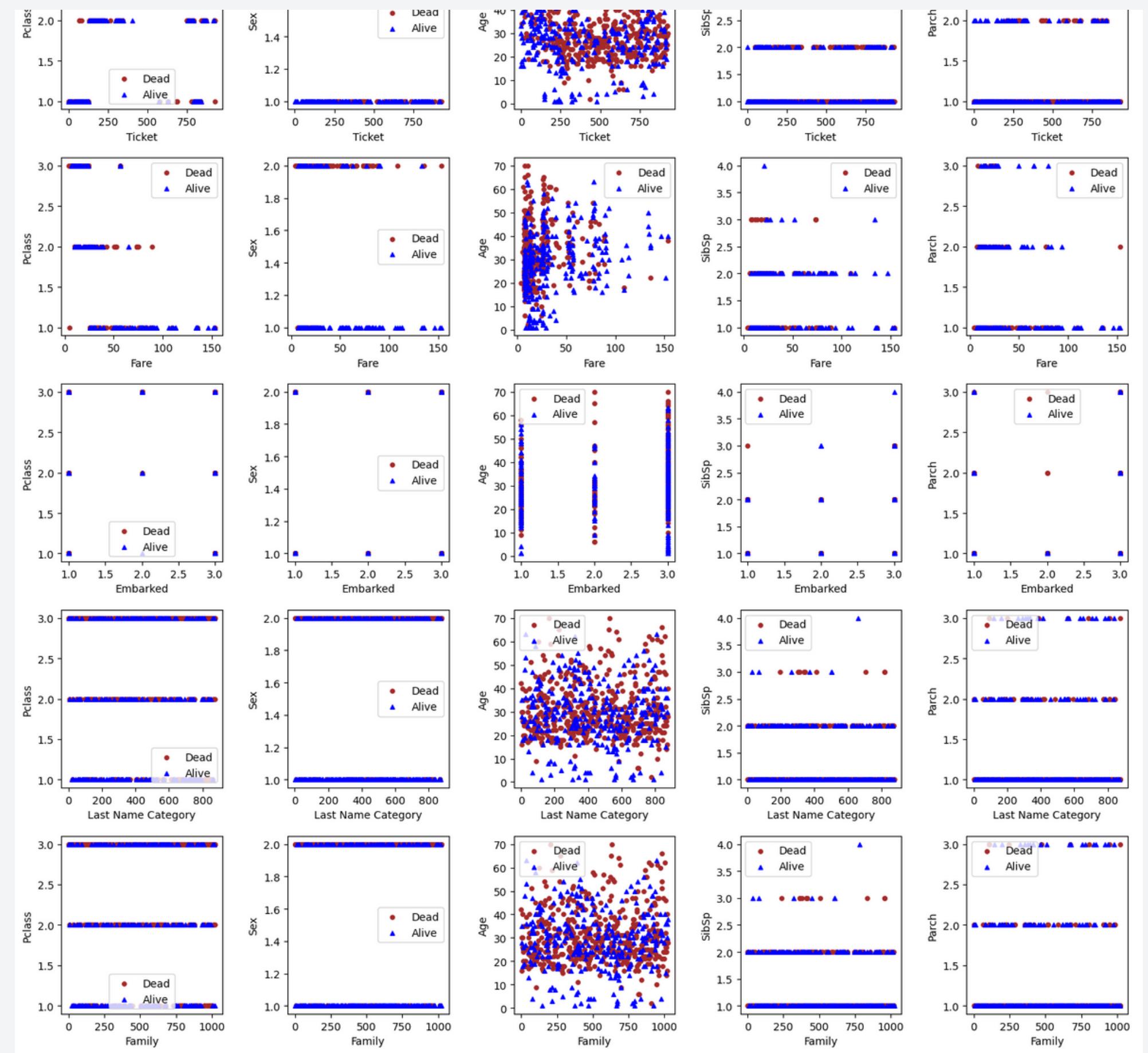












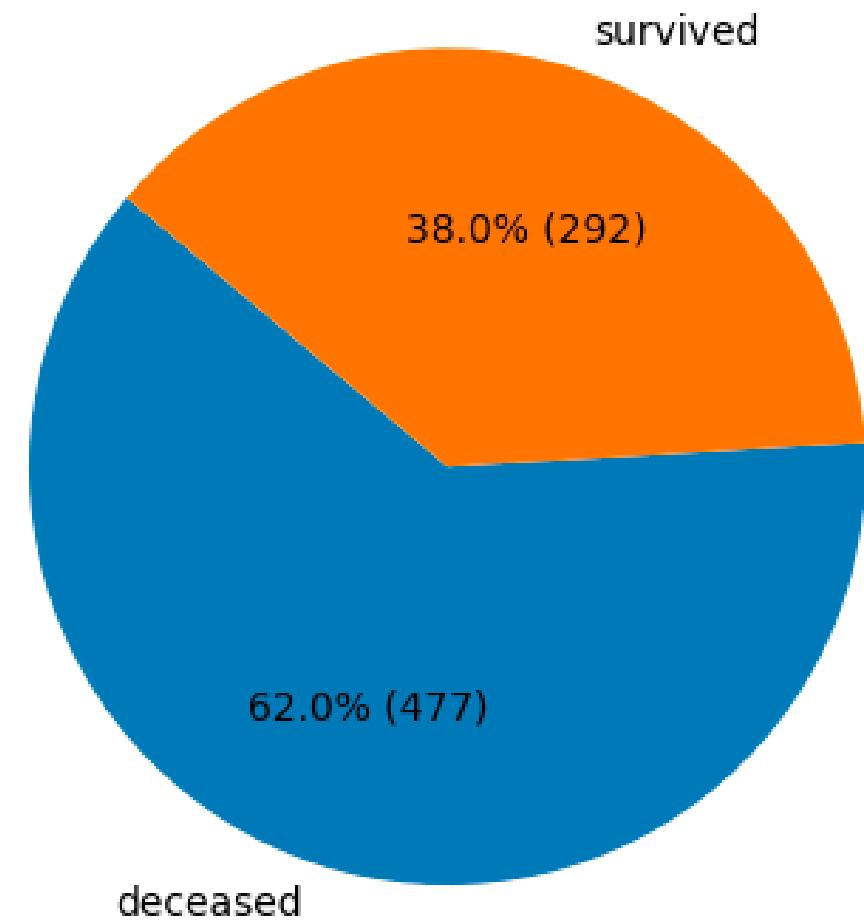
BALANCING

01

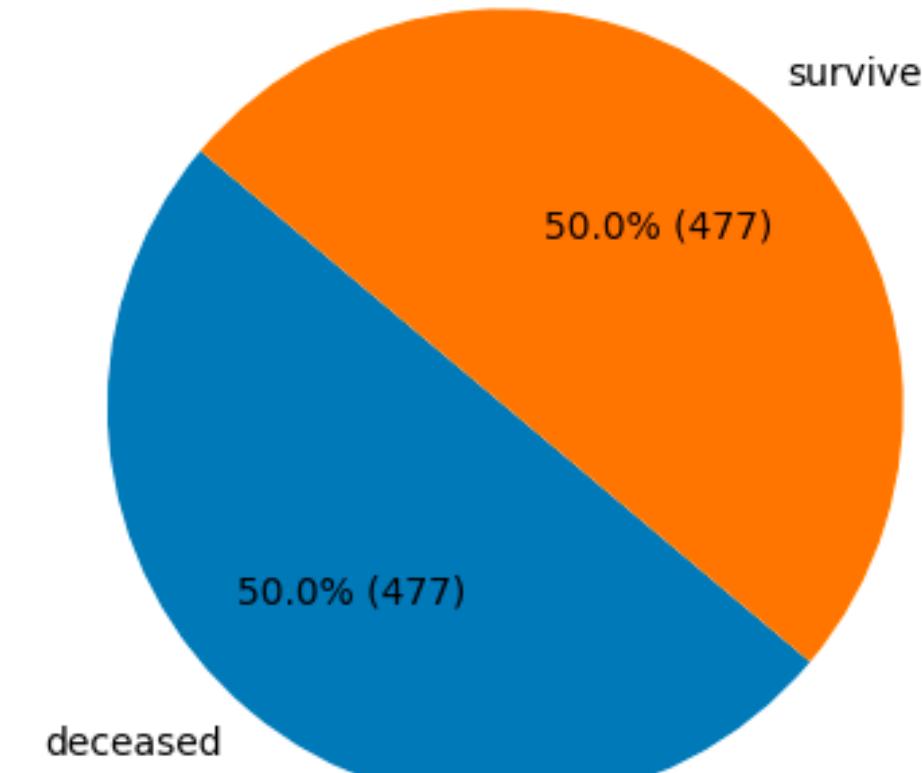
Balanceo del dataframe

Utilizando la libreria de SMOTE, creamos nuevos datos de sorevivientes para tener la misma cantidad de personas fallecidas y obtener mejores resultados

Imbalanced dataset plot



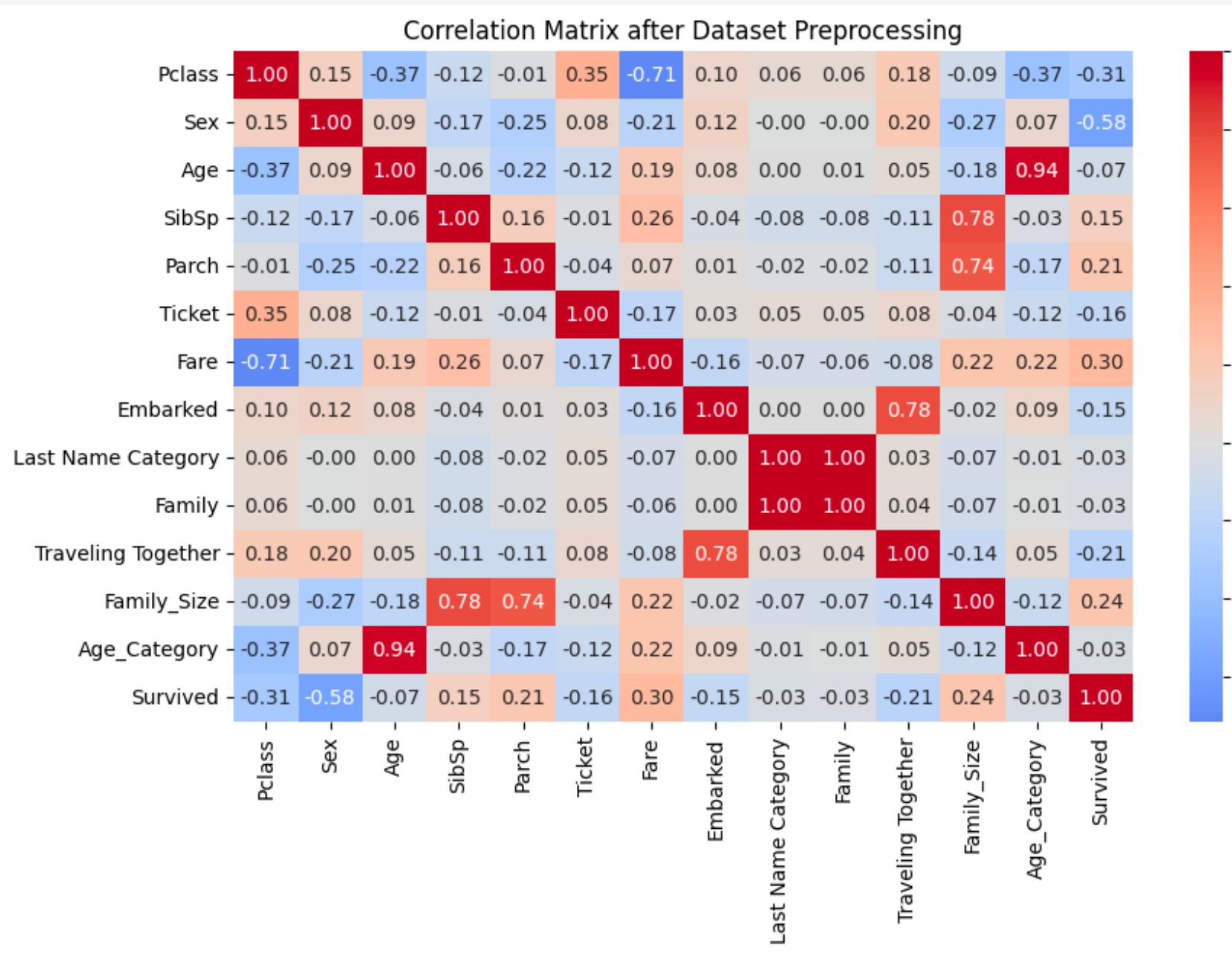
Balanced dataset plot



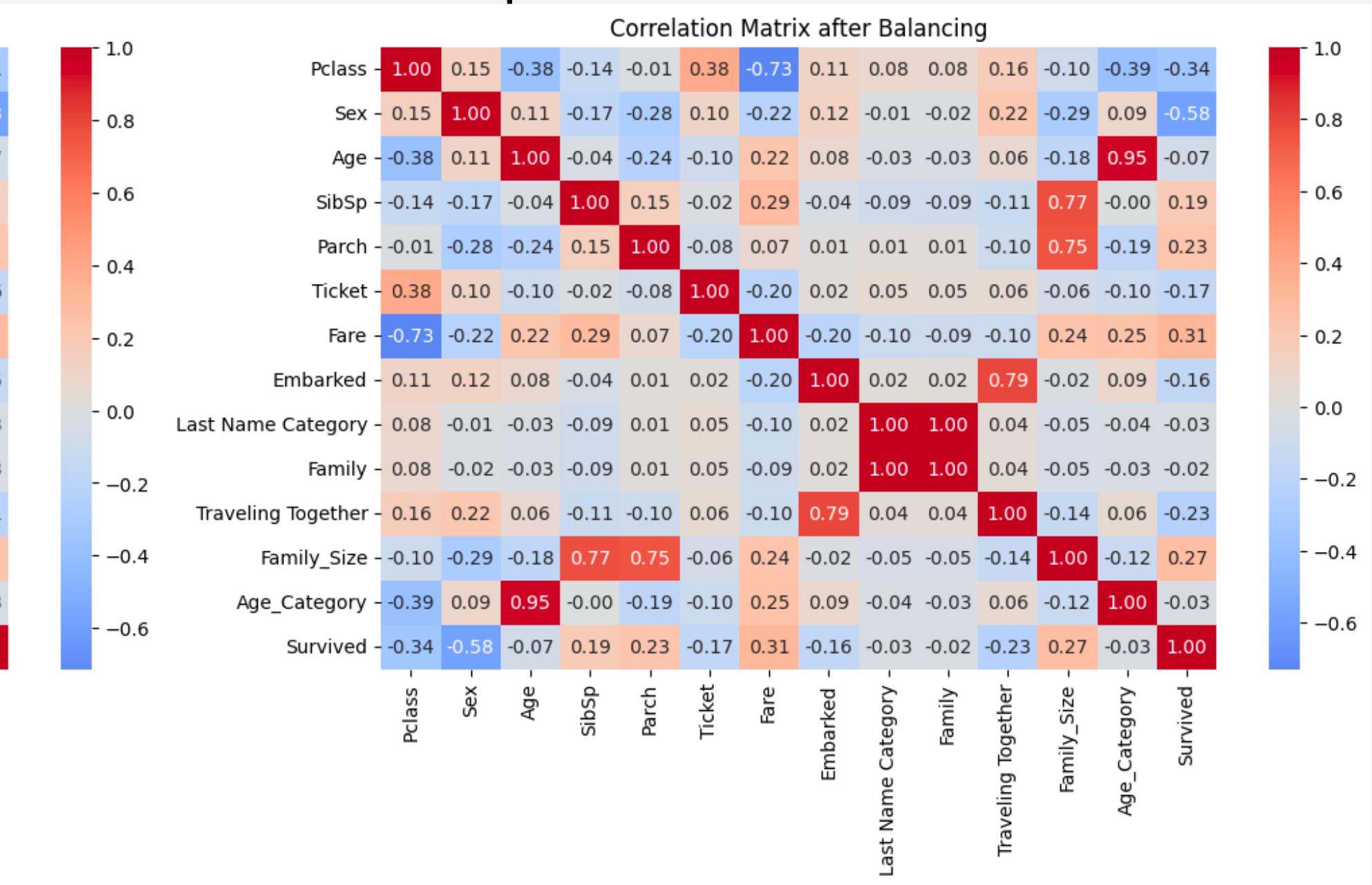
Synthetic Minority Over-sampling

Technique(SMOTE) es una biblioteca de imbalanced-learn para **aumentar la cantidad de la clase minoritaria** en el dataset, esto es posible ya que crea nuevos datos combinando las muestras ya existentes de la clase minoritaria

Antes de balanceo



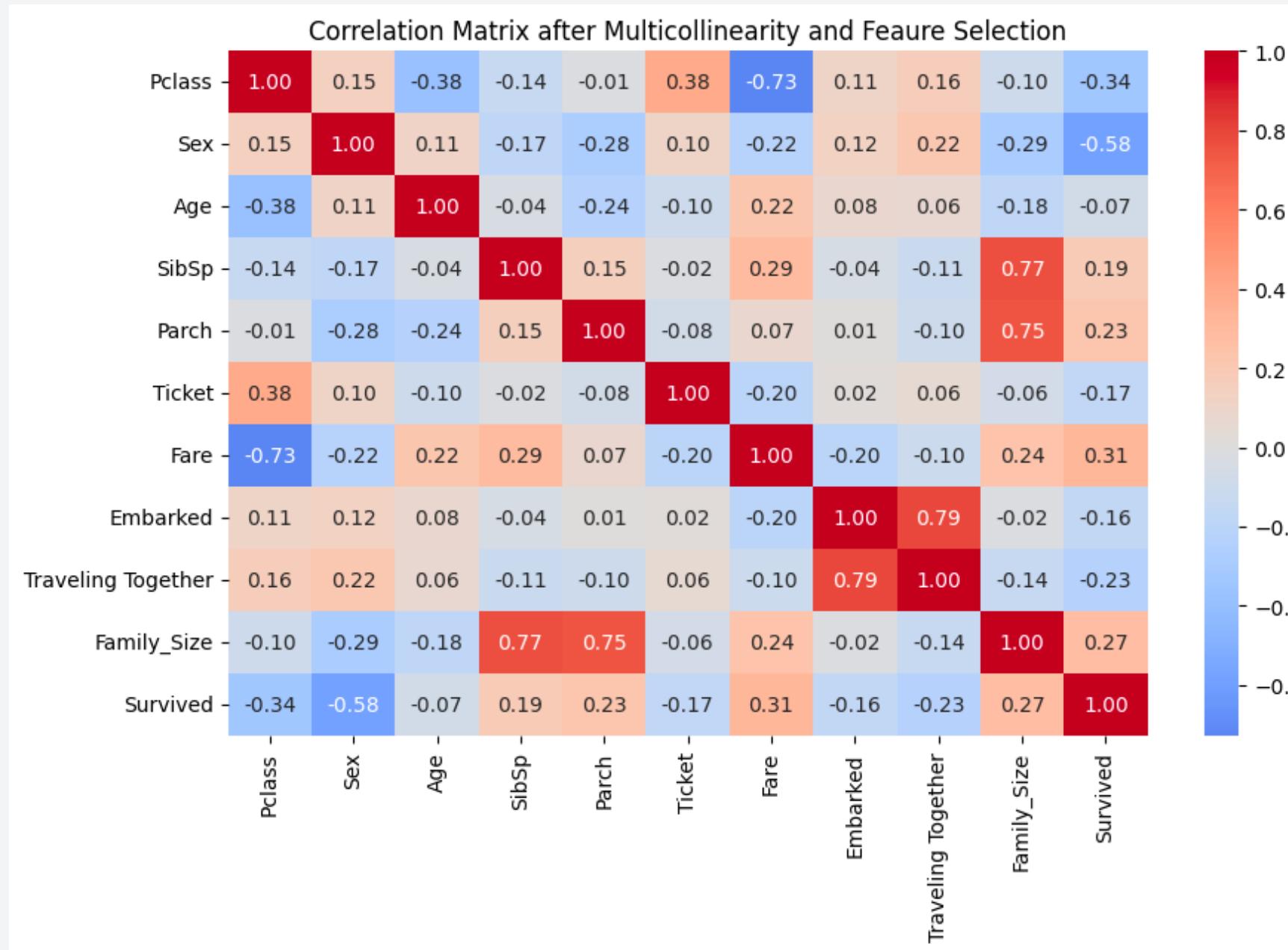
Despues del balanceo



- La **correlación** entre "Sexo" y "Sobrevivió" se vuelve **más negativa**, indicando una **relación inversa más fuerte entre el género y la supervivencia**. Las tasas de supervivencia entre hombres y mujeres se hace más evidente
- La correlación entre "Clase de pasajero" y "Sobrevivió" también se vuelve más negativa, destacando la **influencia de la clase de pasajero en la probabilidad de supervivencia**.

MULTICOLLINEARITY AND FEATURE SELECTION

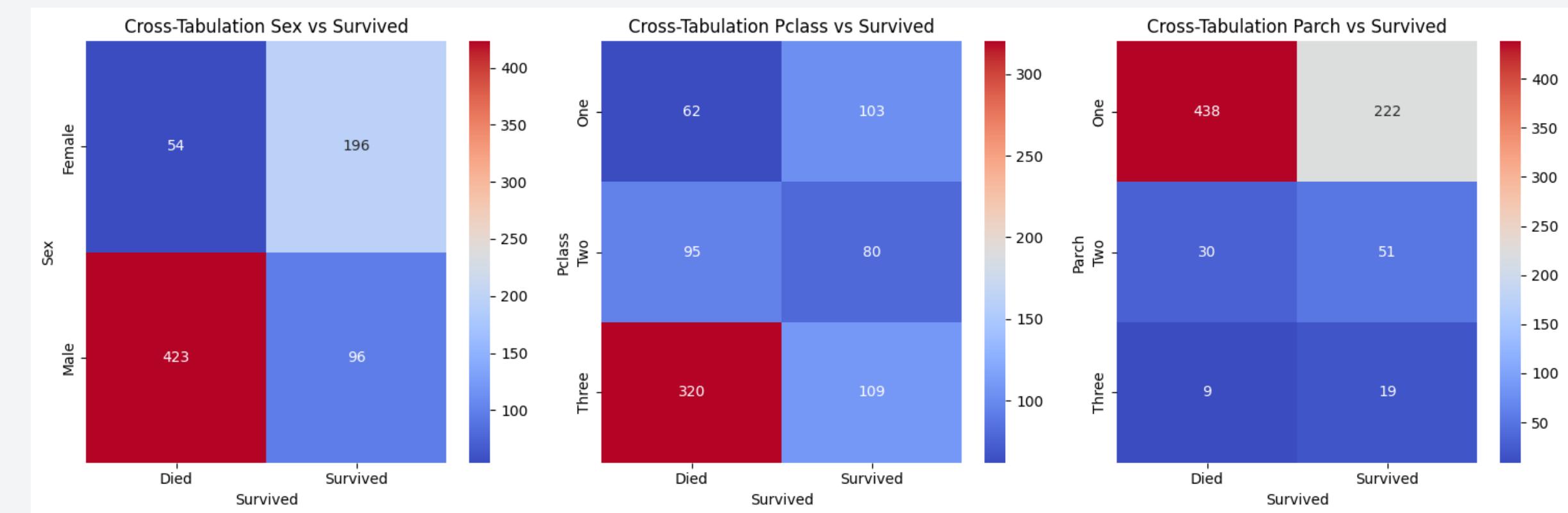
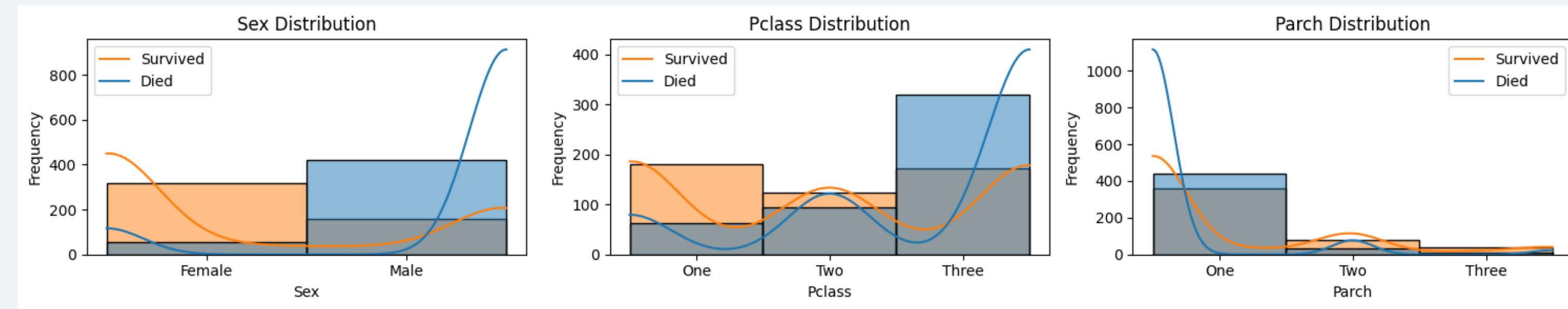
- Establecemos un **umbral para la fuerza** de correlación, en este caso, **0.05**.
- **Iteramos** a través de las **correlaciones**.
- Si la **correlación** absoluta está por **debajo** del **umbral**, **eliminamos** la característica correspondiente.
- Visualizamos la matriz de correlación actualizada para asegurarnos de que las características altamente correlacionadas se hayan eliminado correctamente
- Esto contribuye a un conjunto de características más efectivo para el aprendizaje automático.

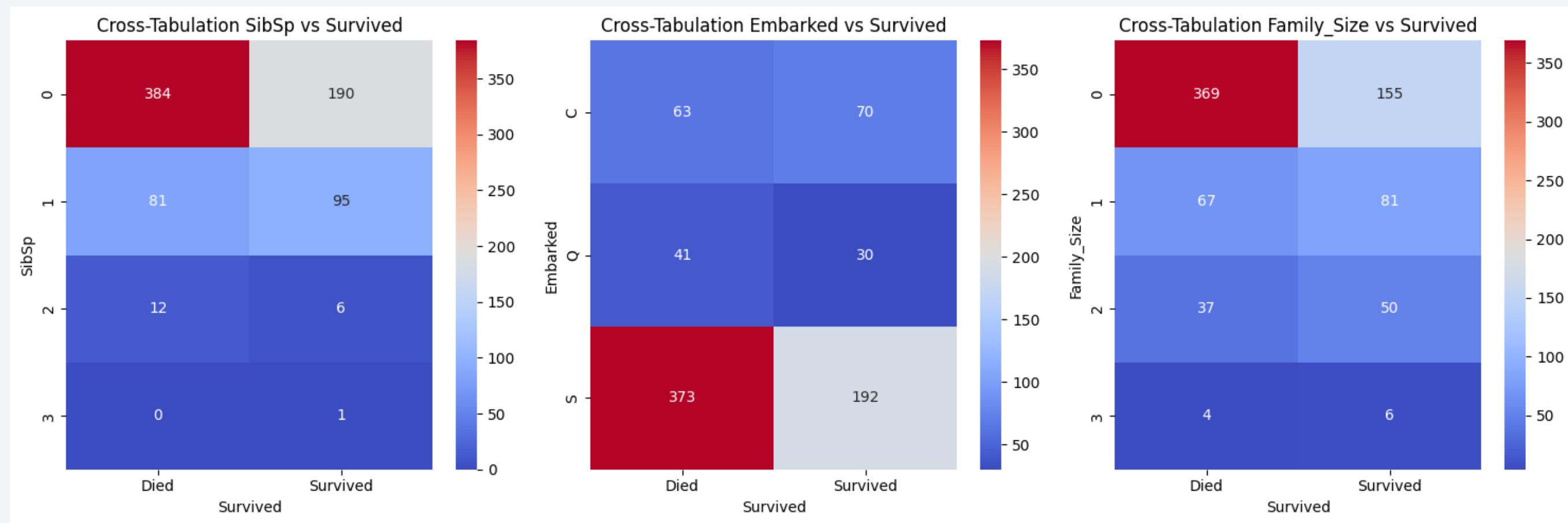
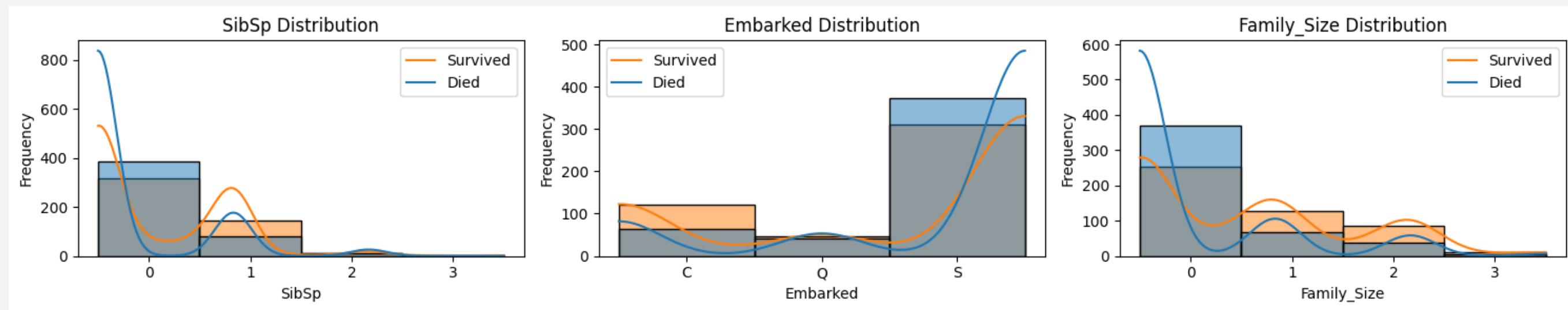


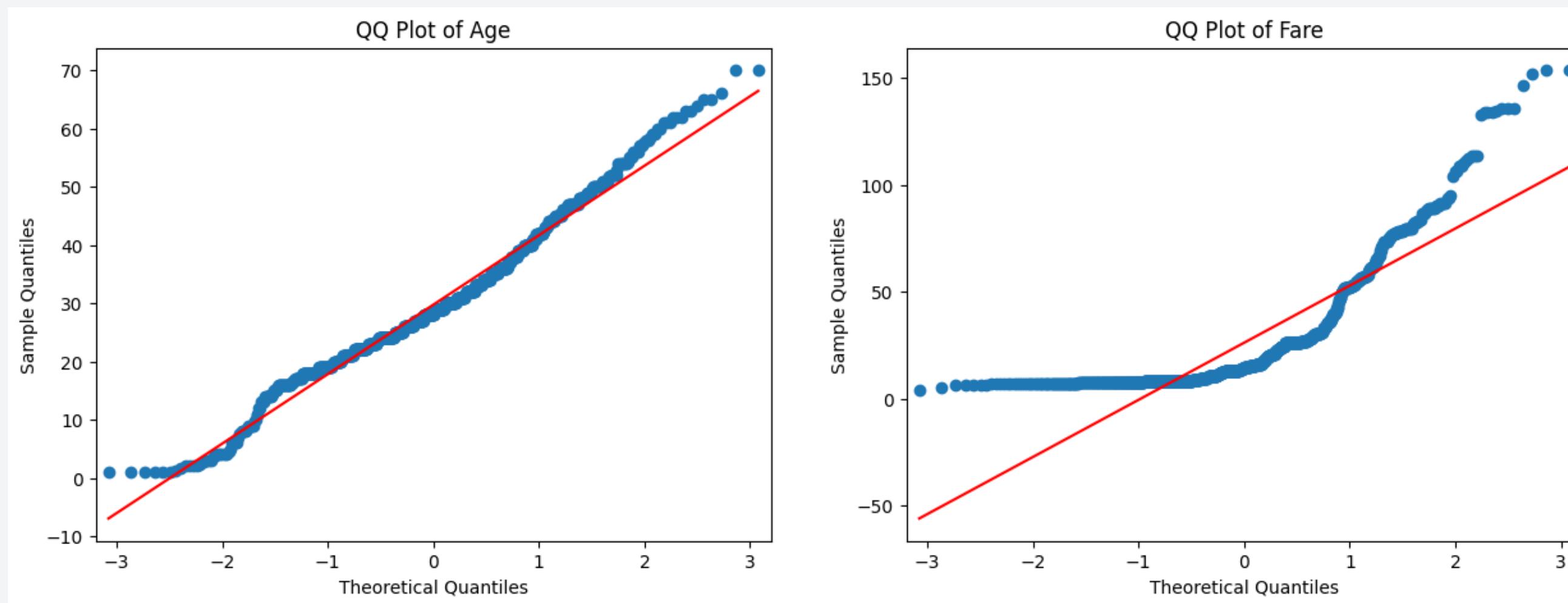
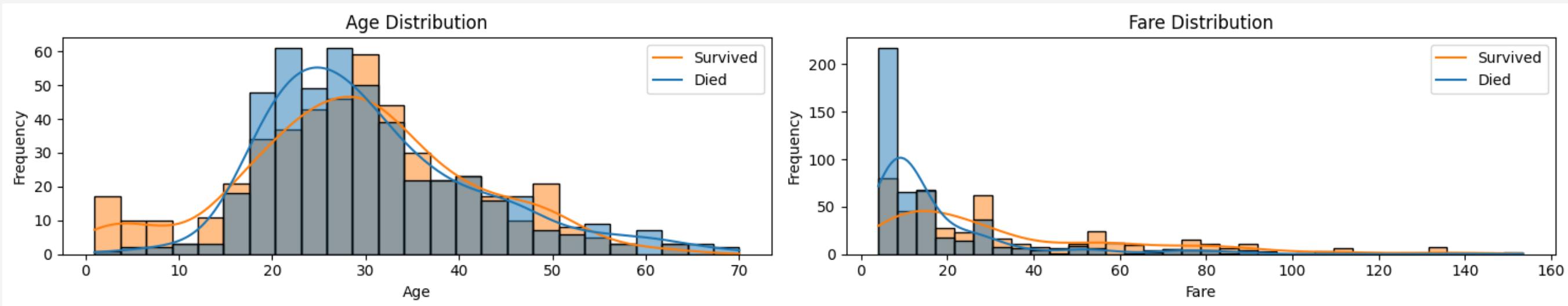
- Las características eliminadas incluyen **"Categoría de Apellido," "Familia," "Categoría de Edad," y "Embarcado."**
- Estas características se consideraron débilmente correlacionadas con la supervivencia y **se eliminaron para mejorar la eficiencia del modelo y reducir el ruido.**
- Resulta en un conjunto de **características más enfocado con relaciones más sólidas** con la variable objetivo, lo que puede mejorar el rendimiento predictivo de los modelos de aprendizaje automático.

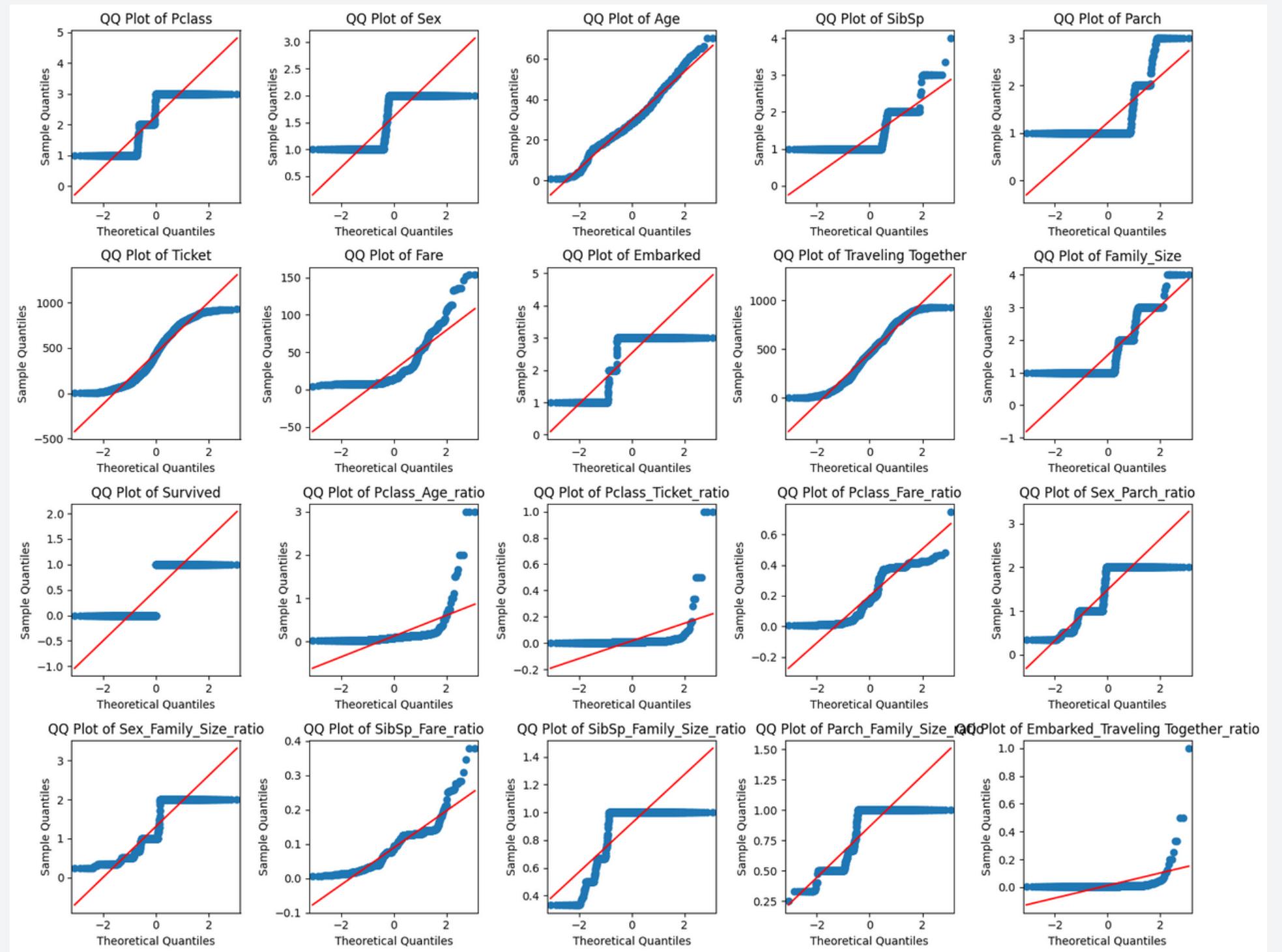
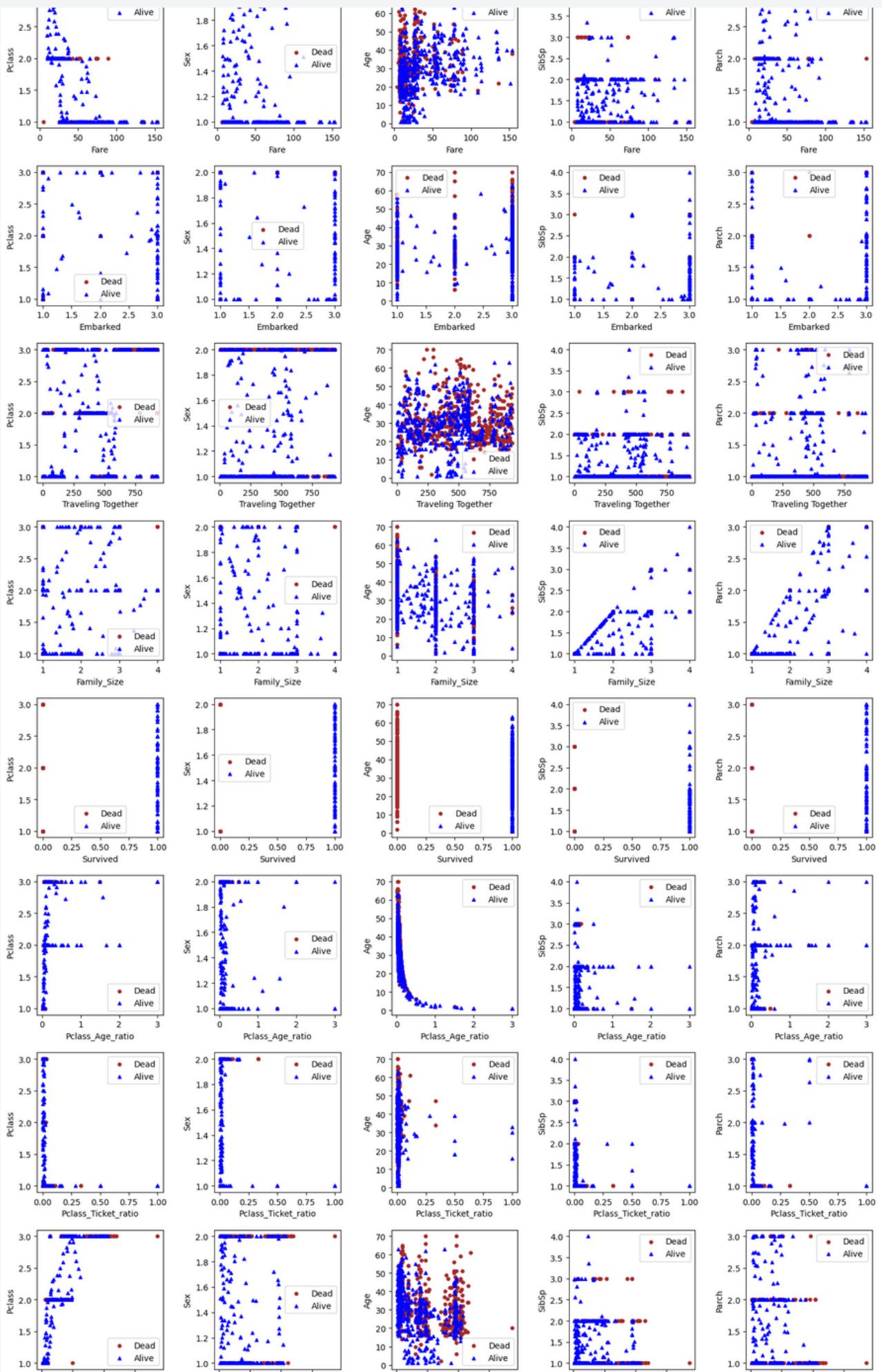
RATIOS

- Identificamos pares de columnas dentro del conjunto de datos con un **coeficiente de correlación mayor o igual a 0.25** o menor o igual a **-0.25**.
- Creamos nuevas características de proporción para cada par identificado.
- Hemos introducido varias características de proporción nuevas en el conjunto de datos.
- Por ejemplo, "**Pclass_Ticket_ratio**" representa la proporción entre "Pclass" y "Ticket," "**Sex_Fare_ratio**" representa la **proporción entre "Sexo" y "Tarifa,"** y así sucesivamente.
- Estas características de proporción proporcionan información adicional que puede capturar patrones o interacciones significativas dentro de los datos.









SCALING OR NORMALIZING

Codificación de Características Categóricas:

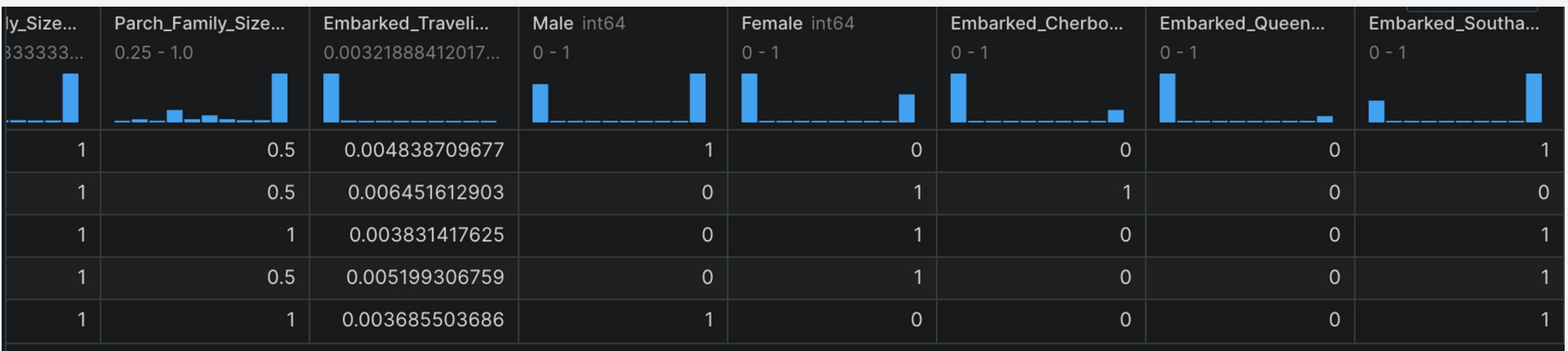
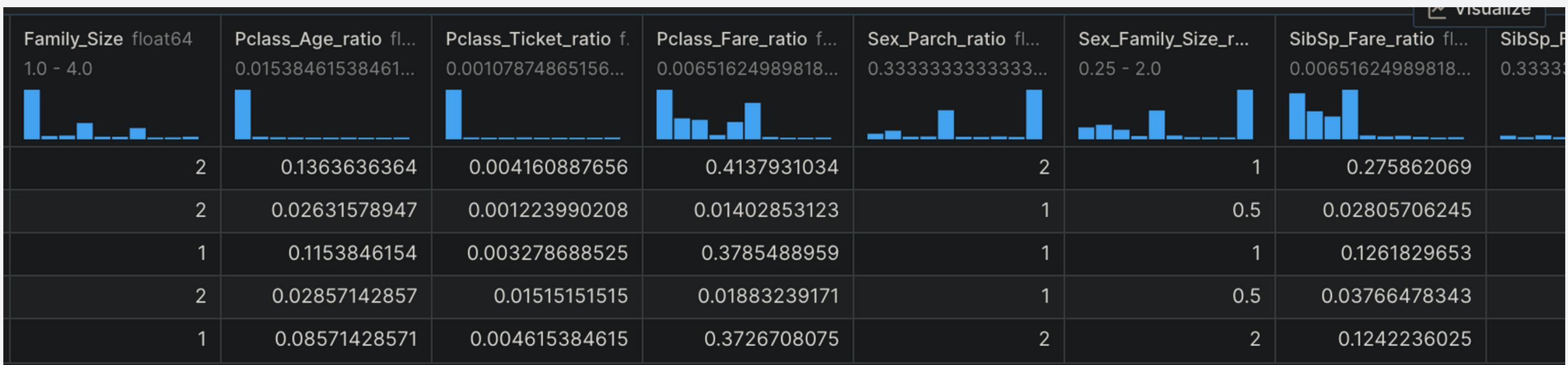
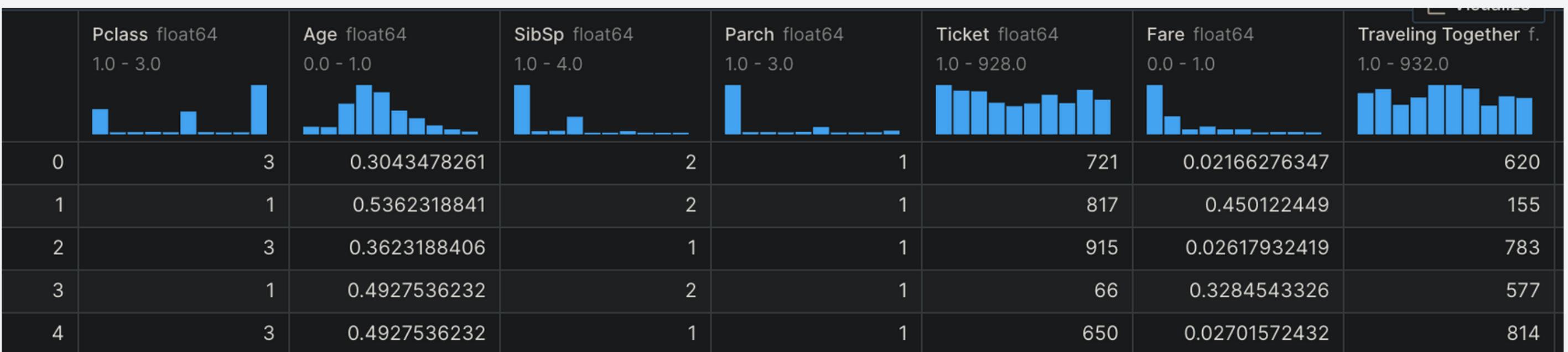
- Específicamente:
 - Creamos las columnas binarias 'Hombre' y 'Mujer' basadas en la característica 'Sexo', convirtiéndola en un formato codificado one-hot.
 - Creamos las columnas binarias 'Embarque_Cherburgo', 'Embarque_Queenstown' y 'Embarque_Southampton' basadas en la característica 'Embarque', aplicando codificación one-hot.

Escalado y Normalización de Características Numéricas:

- Para características numéricas continuas como 'Edad' y 'Tarifa', optamos por normalizarlas mediante escalado Min-Max. Esta transformación escala los datos a un rango específico, generalmente entre 0 y 1, preservando la distribución original.
- Las características numéricas discretas como 'Clase', 'SibSp', 'Parch', 'Categoría de Apellido', 'Familia' y varias características de proporción se dejan sin cambios porque ya están en una escala discreta y no es necesario aplicarles un escalado o normalización adicional.

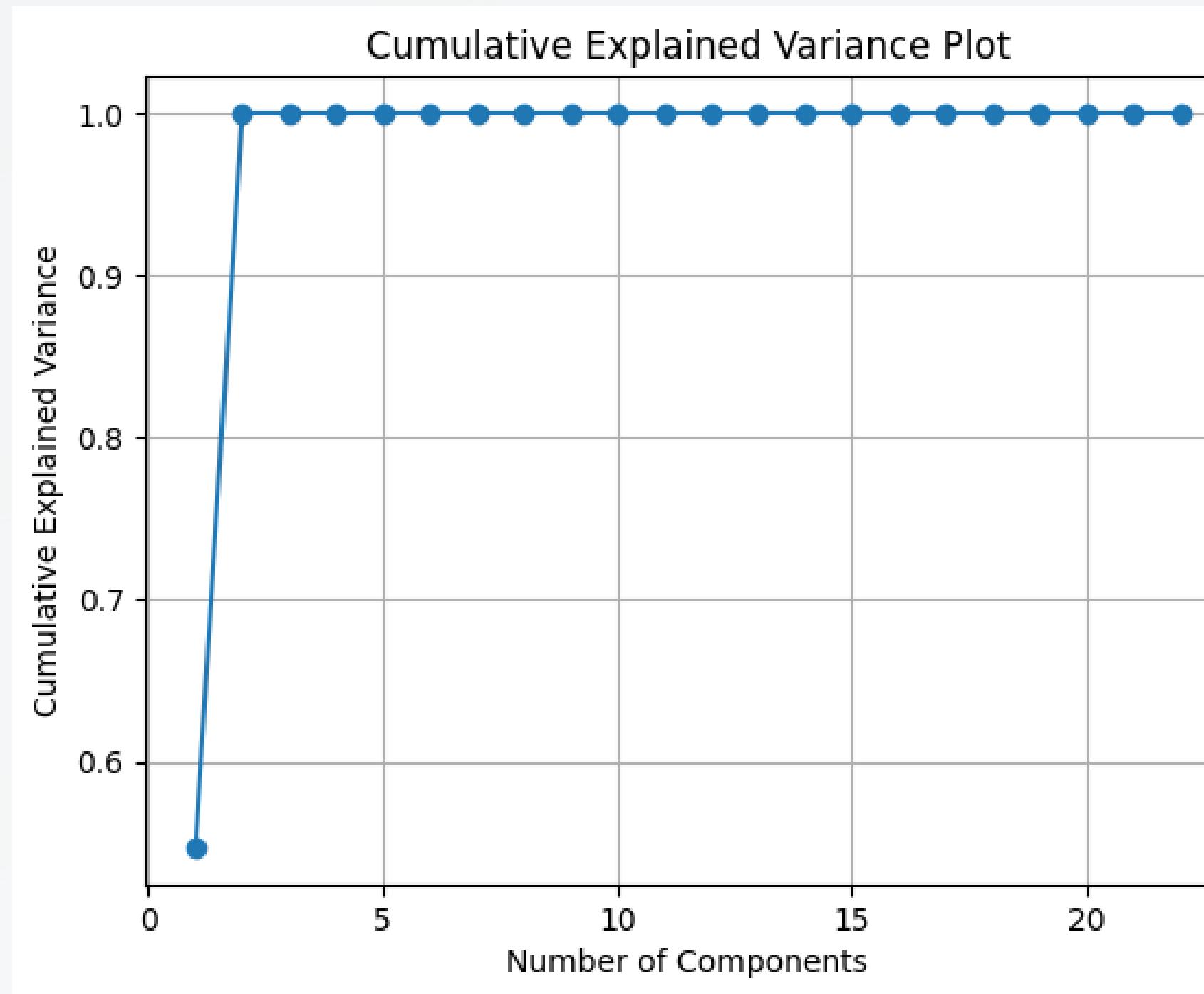
DATASET FINAL

22 features



DIMENSIONALITY REDUCTION

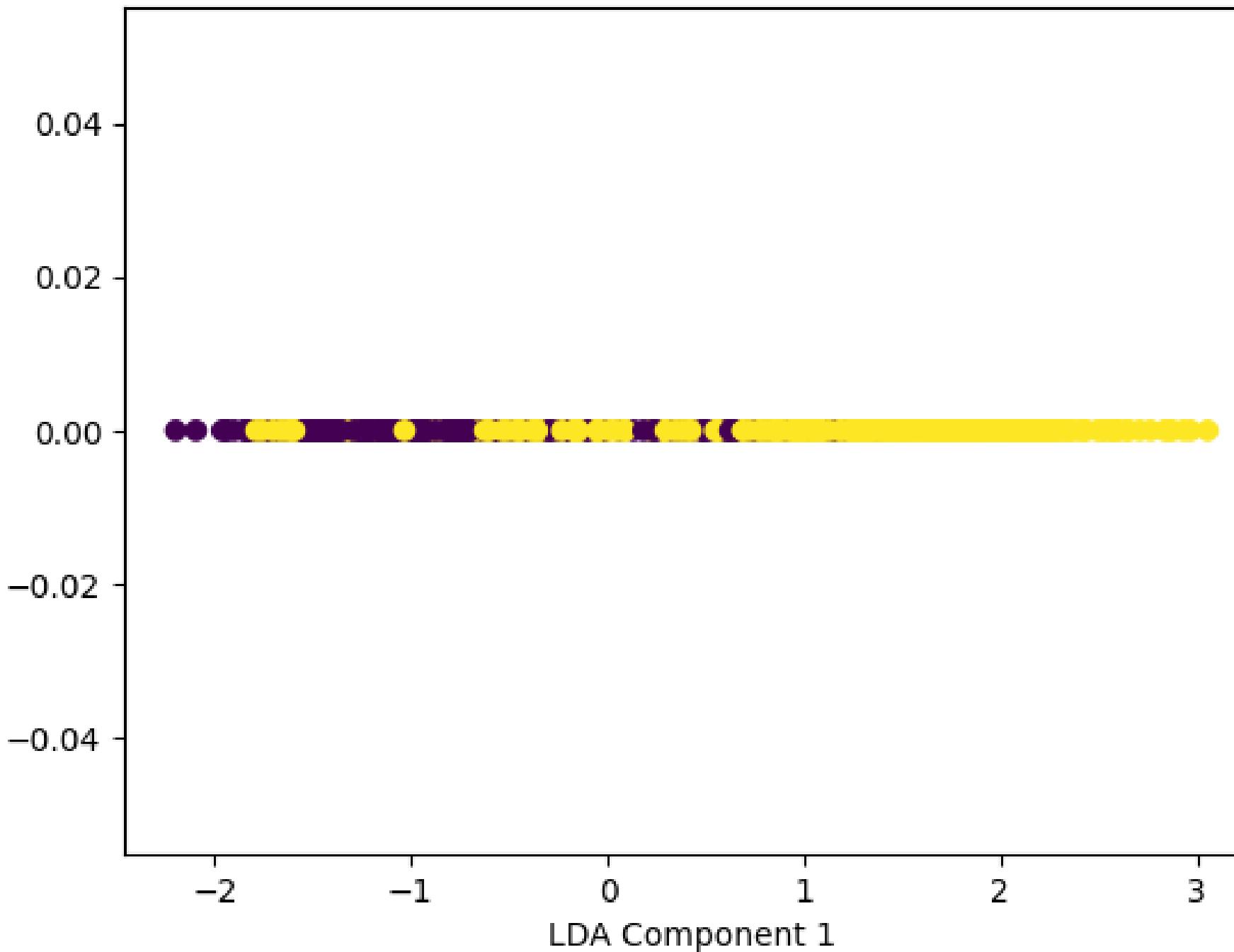
PCA



Ticket	0.944598
Traveling Together	0.328225
Pclass	0.001179
Sex_Family_Size_ratio	0.000362
Embarked_Southampton	0.000346
Embarked_Cherbourg	-0.000345
Sex_Parch_ratio	0.000335
Male	0.000309
Family_Size	-0.000264
Female	-0.000244

LDA

One-Dimensional LDA Projection



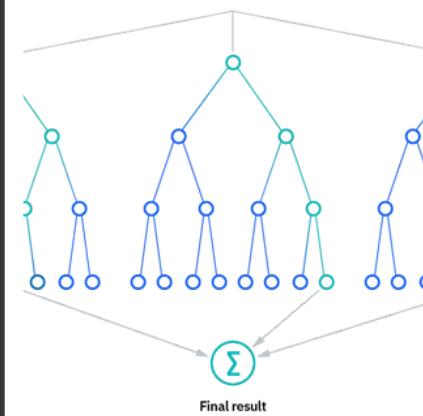
CLASIFICACIÓN

PROCEDIMIENTO GENERAL

- Utilizamos Grid Search para hacer una comparacion de todas las posibles combinaciones de los hiperparametros que elegimos y de esta manera obtener los mejores parametros.
- Utilizamos los mejores parametros que obtuvimos para entrenar nuestro modelo y poder realizar el Train-test-split de nuestra data y obtener de esta manera nuestra matriz de confusión

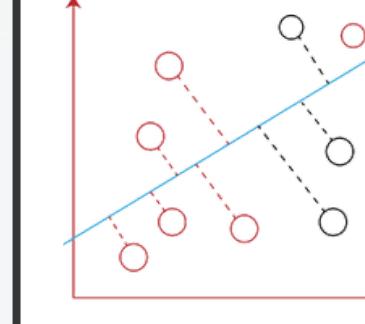
SELECCIÓN DE CLASIFICADORES

Random forest



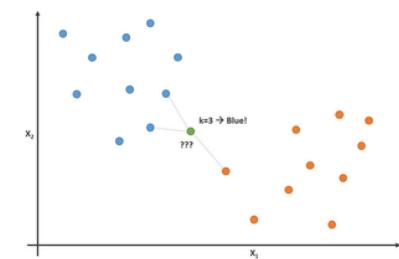
Se optó por utilizar Random Forest debido a su capacidad de contar con múltiples estimadores que el usuario puede ajustar según sea necesario. Esta flexibilidad permite adaptar el modelo de manera más versátil a diferentes requerimientos y situaciones

LDA



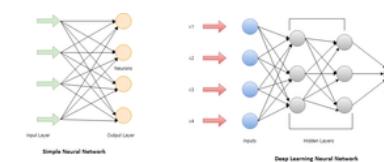
Es una herramienta de modelado simple y efectiva que se destaca por su capacidad para presentar resultados de manera clara y comprensible. Además, LDA ofrece una función adicional muy valiosa: la capacidad de reducir la dimensionalidad de los datos, simplificando así la representación de la información de manera más concisa

K-Nearest Neighbors



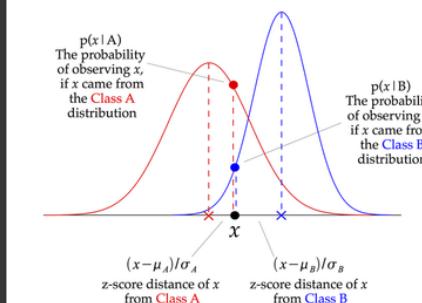
Se propuso por la capacidad que tiene para adaptarse a los datos. Ya sea por que las características no tiene estructura o por ser un conjunto mediano. También por forma de visualización la cual es de fácil entendimiento

Binary feedforward network



Este modelo se seleccionó principalmente como un experimento para observar cómo los datos se comportarían con un enfoque neuronal.

GaussianNB



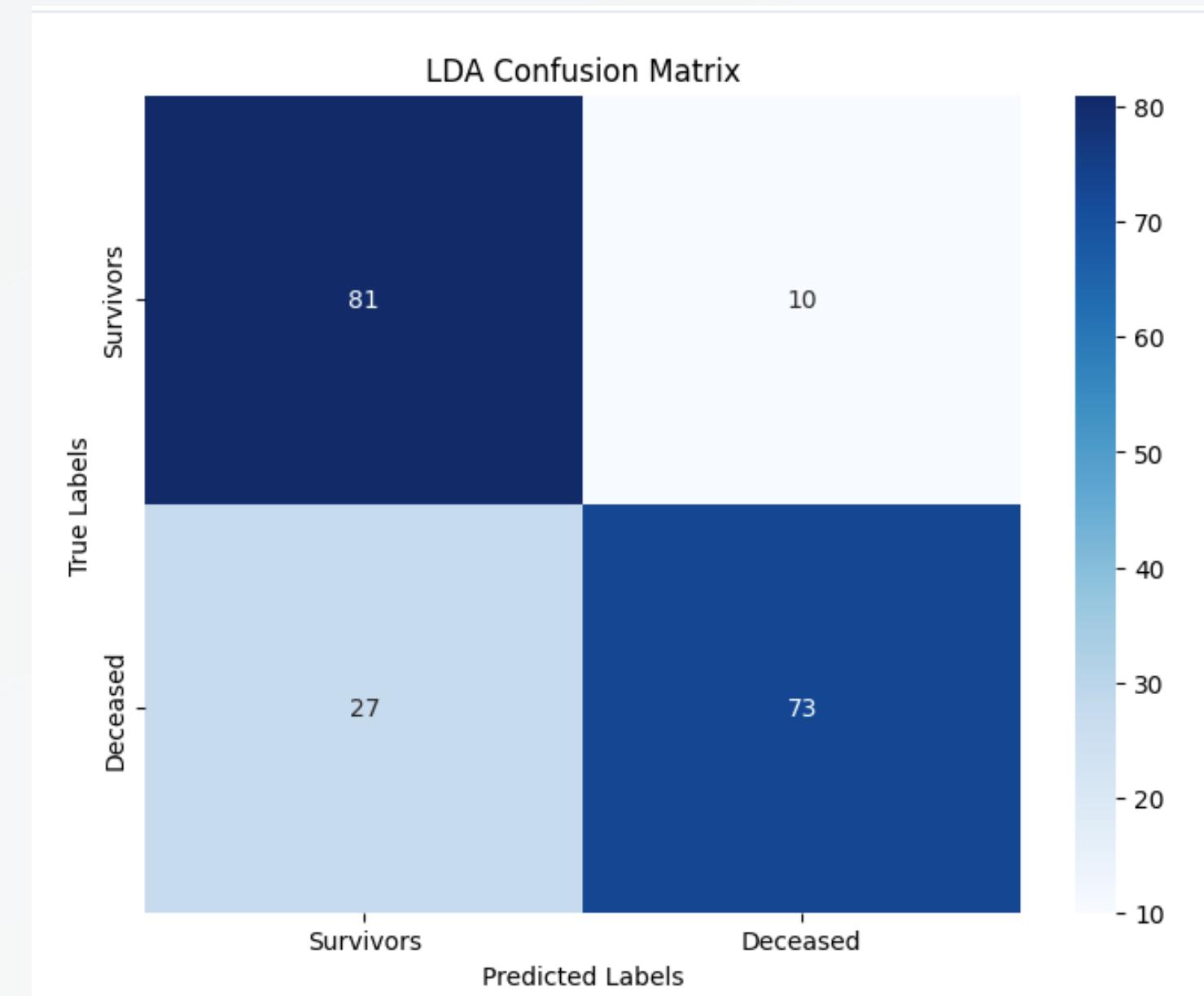
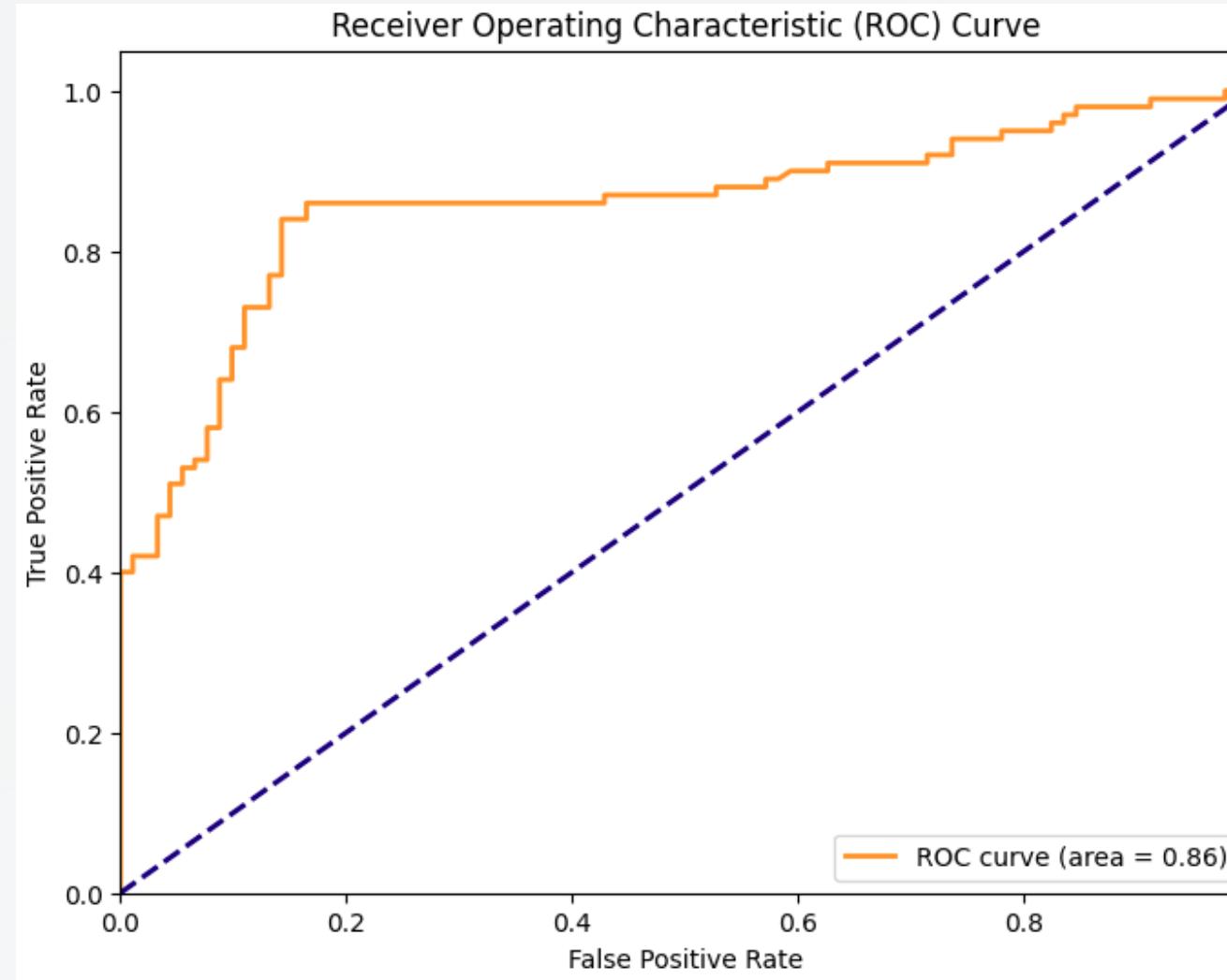
Se eligió por su capacidad de suposición de independencia entre las características. Aunque tenemos muchas características que claramente dependen de una o más características (como las proporciones), estamos apostando por los componentes LDA.

MÉTRICAS DE EVALUACIÓN -> LDA

{'shrinkage': 'auto', 'solver': 'lsqr'} - AggP: 0.8197519977955359

Best aggregate performance: 0.82

Accuracy score: 0.81
Survivors precision: 0.75
Survivors recall: 0.89
Deceased precision: 0.88
Deceased recall: 0.73



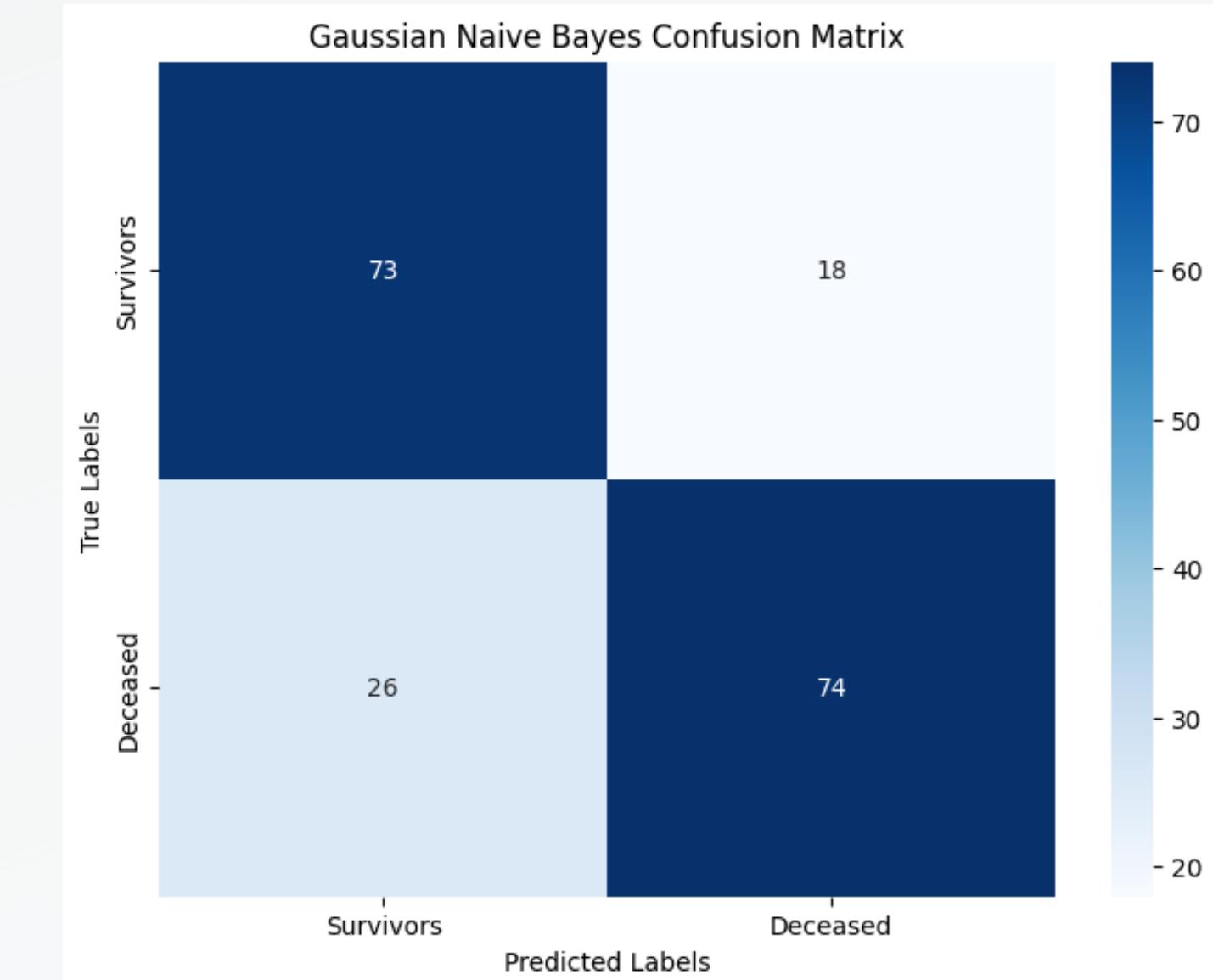
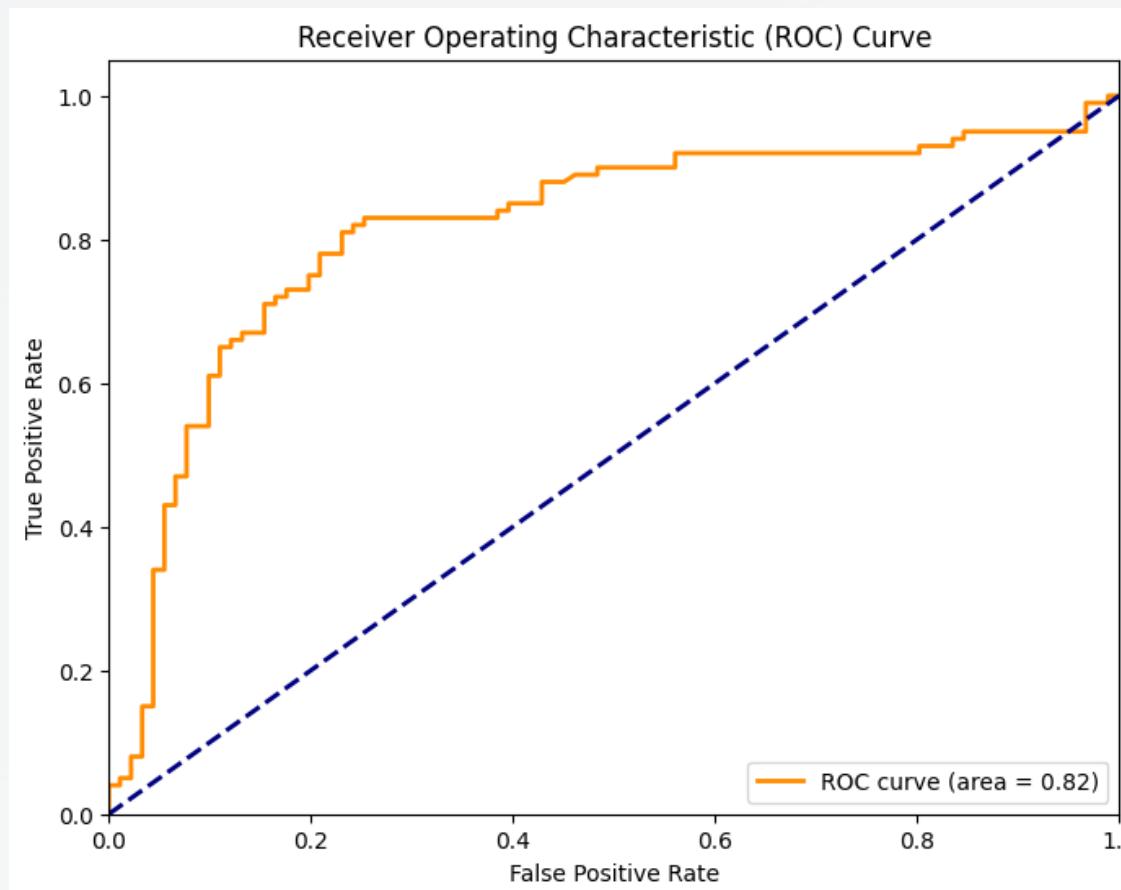
The accuracy score of 0.81 suggests a reasonably accurate classification. However, the precision for survivors 0.75 and deceased 0.88 indicates moderate to high bias, which means that there might be a trade-off between correctly identifying survivors and deceased passengers. Additionally, the recall for survivors (0.89) is relatively high, while the recall for deceased 0.73 is moderate, indicating a moderate level of variance in the model's ability to capture true positive cases. These results suggest a good fit without significant underfitting or overfitting.

MÉTRICAS DE EVALUACIÓN -> GAUSSIANNB

{'priors': [0.62, 0.37], 'var_smoothing': 1e-06}

Best aggregate performance: 0.79

Accuracy score: 0.77 Survivors
precision: 0.74 Survivors recall: 0.80
Deceased precision: 0.80 Deceased
recall: 0.74

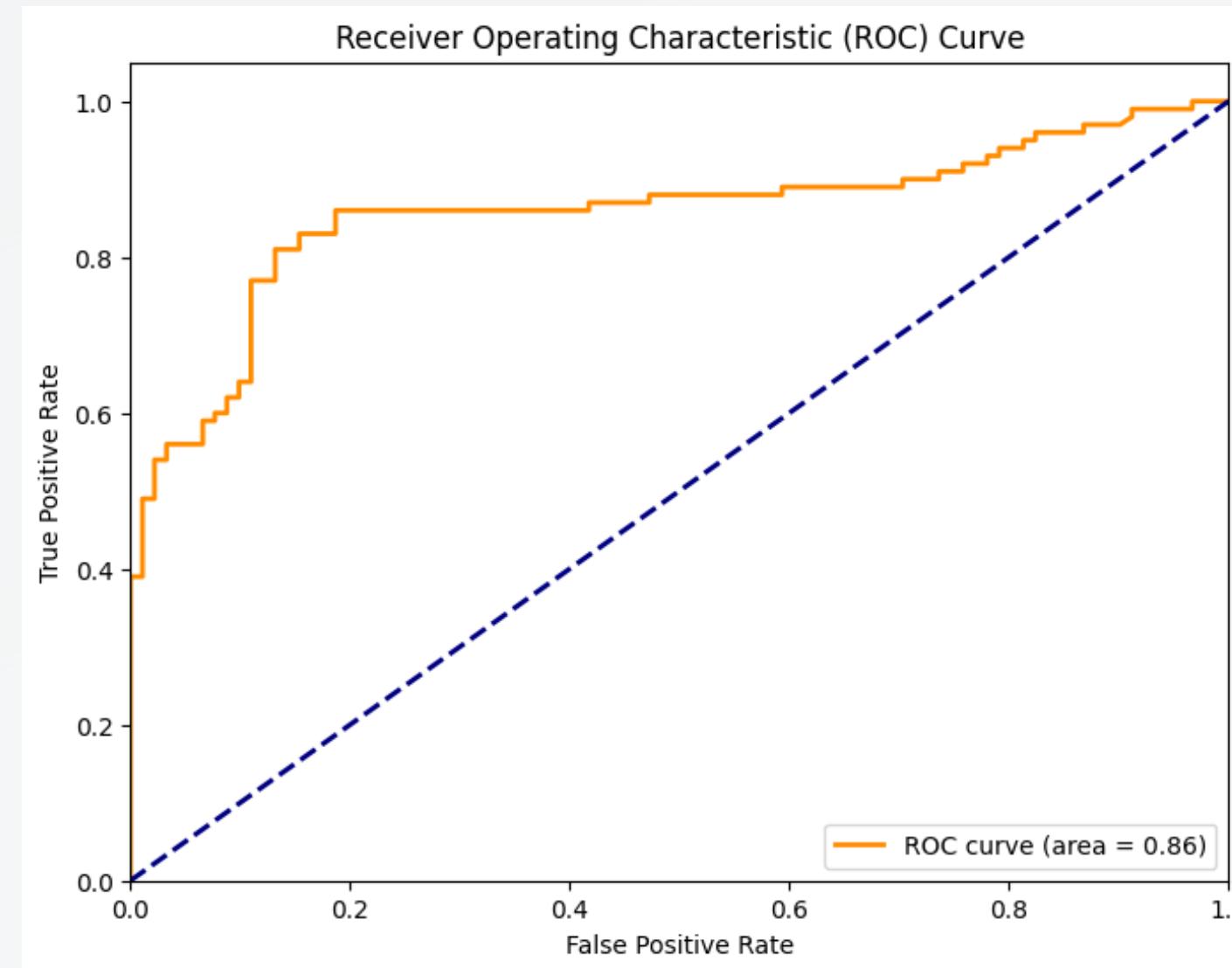


The accuracy score of 0.84 indicates a reasonable classification. The precision scores suggest a balanced model with relatively low bias, identifying most cases. The recall scores indicate a good balance between capturing true positives, which demonstrate a moderate level of variance. The model achieves a good fit with a small amount of underfitting / overfitting, to strike a balance between bias and variance.

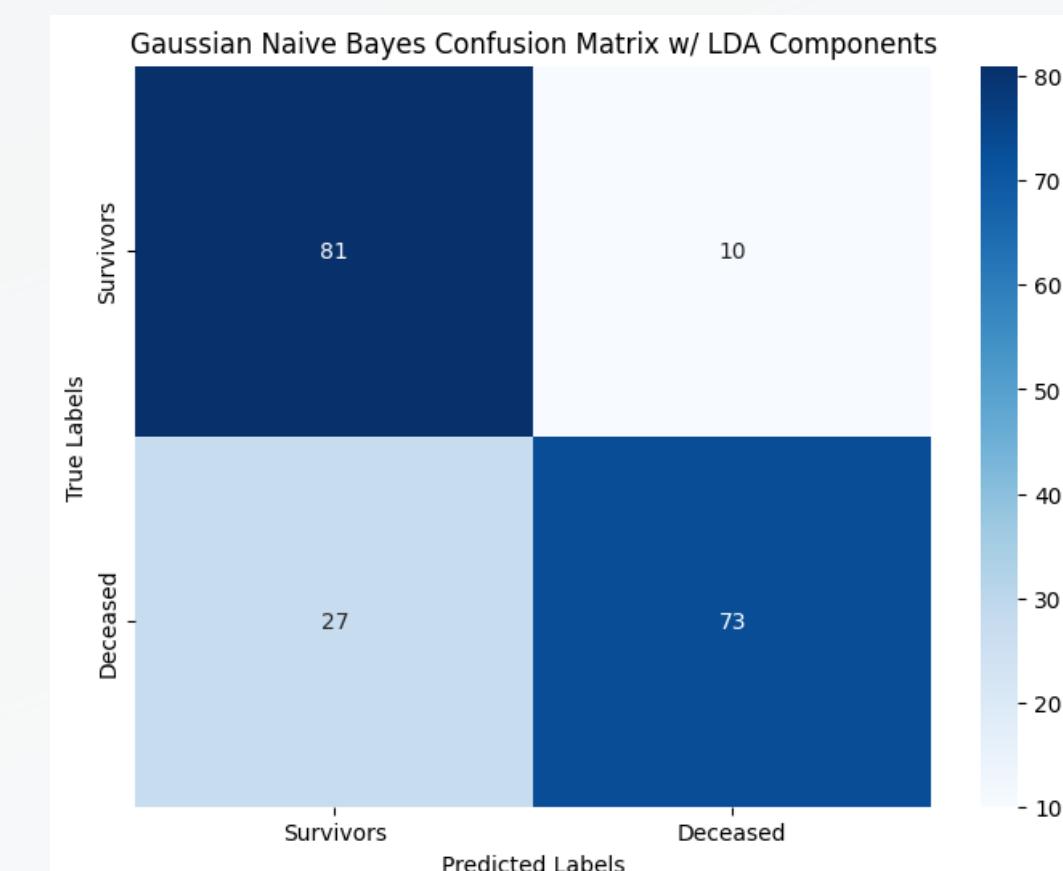
MÉTRICAS DE EVALUACIÓN -> GAUSSIANNB LDA

{'priors': [0.62, 0.37], 'var_smoothing': 1e-09}

Best aggregate performance: 0.83



Accuracy score: 0.81
Survivors precision: 0.75
Survivors recall: 0.89
Deceased precision: 0.88
Deceased recall: 0.73

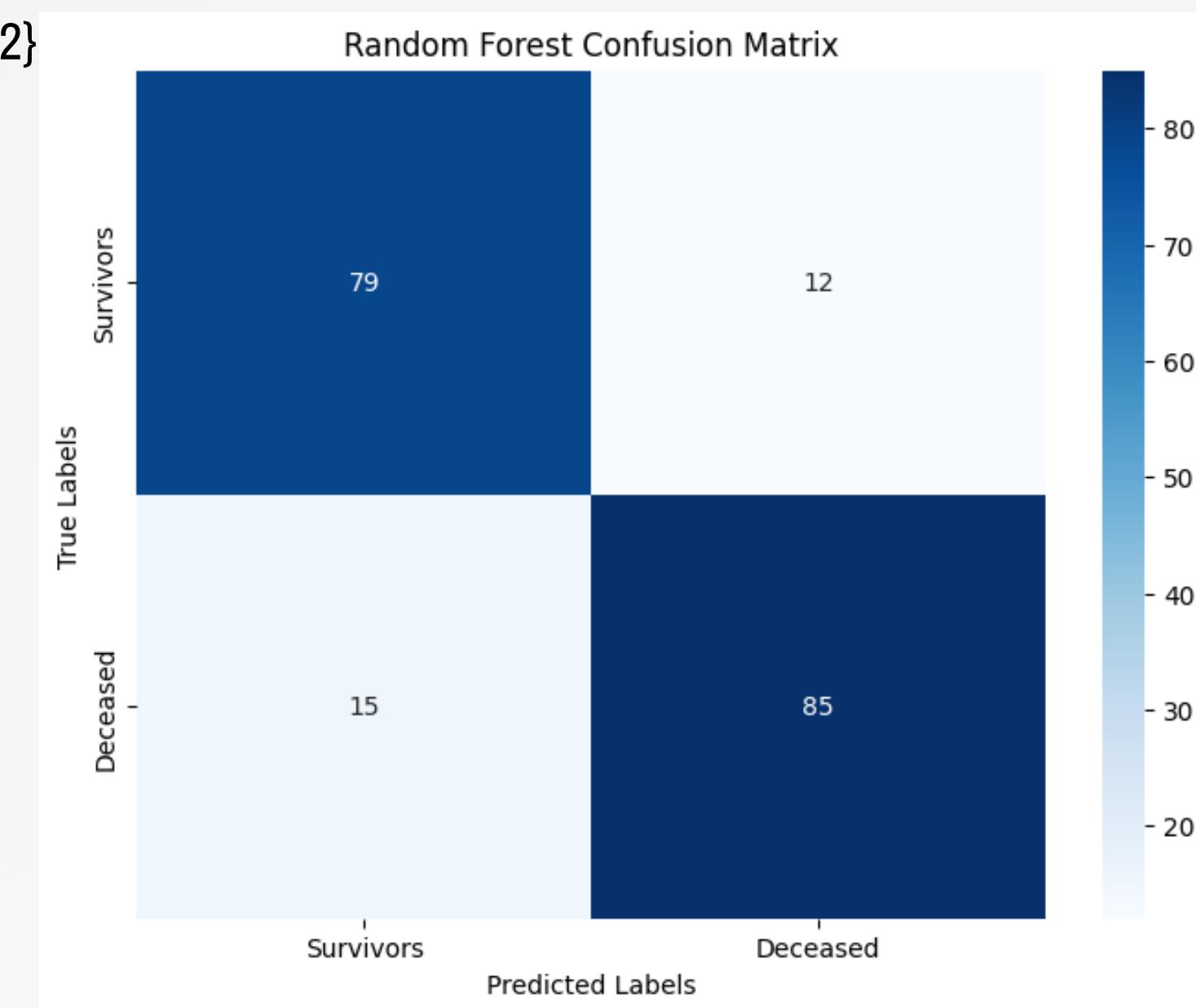
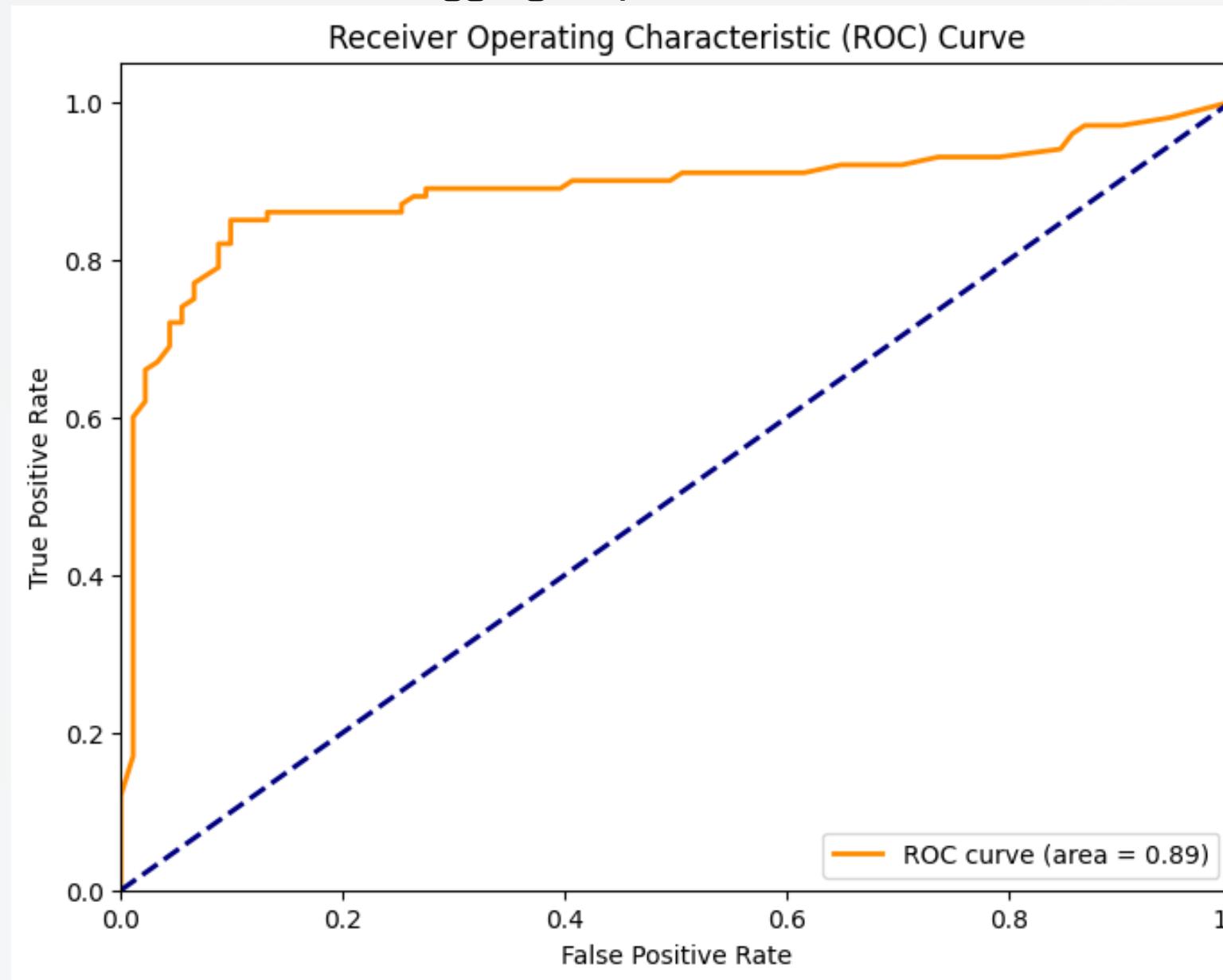


When evaluating the selected GNB model with the best var_smoothing parameter (1e-09) on the test data, it achieved an accuracy score of 0.81, indicating its effectiveness. The confusion matrix analysis shows that the model is somewhat more balanced than previous models, with precision scores of 0.76 for survivors and 0.88 for deceased passengers, along with recall scores of 0.89 for survivors and 0.74 for deceased passengers. It does, however, still hold a significant bias towards survivors.

MÉTRICAS DE EVALUACIÓN -> RANDOM FOREST

{'bootstrap': True, 'max_depth': 20, 'n_estimators': 200, 'random_state': 42}

Best aggregate performance: 0.86



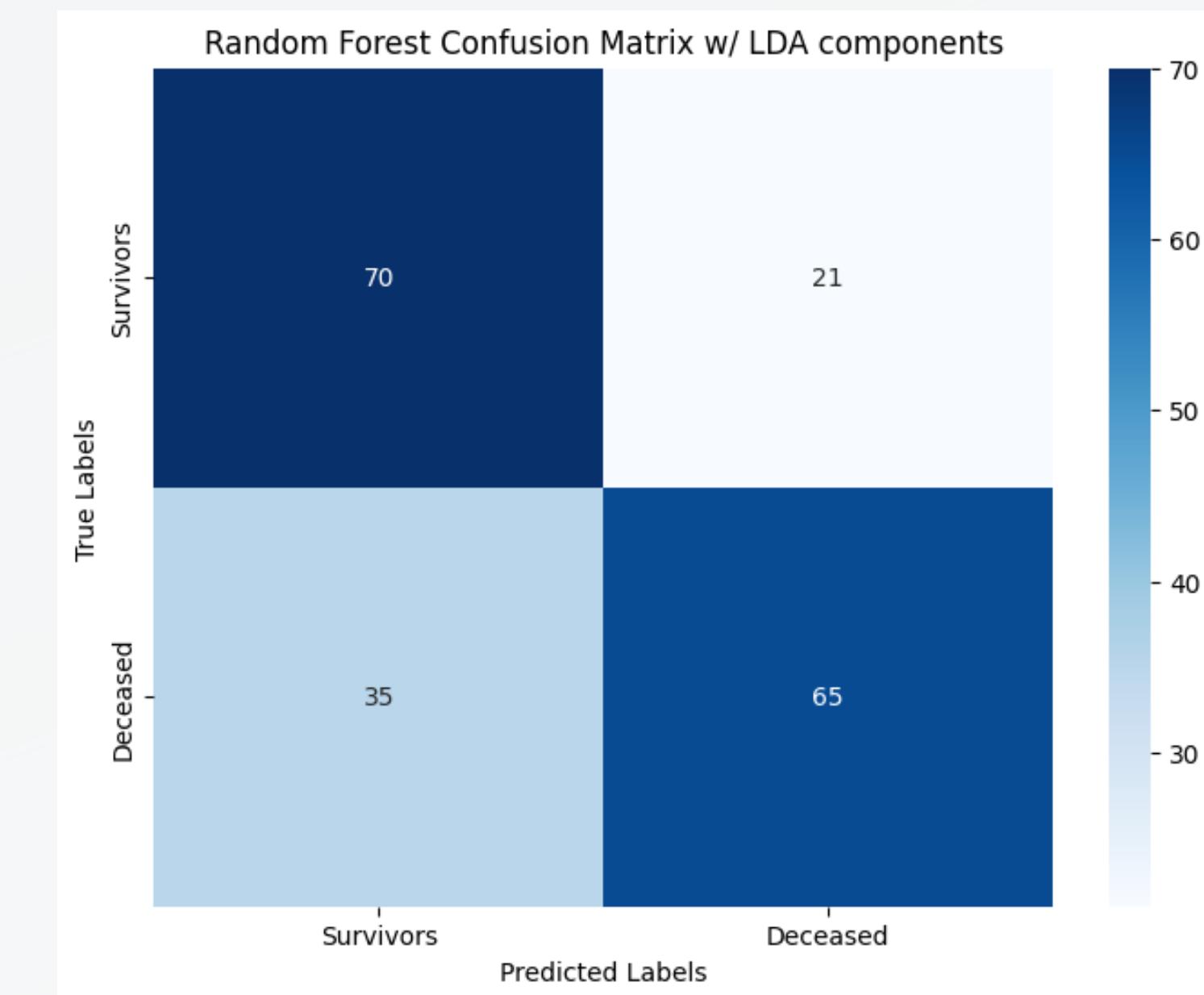
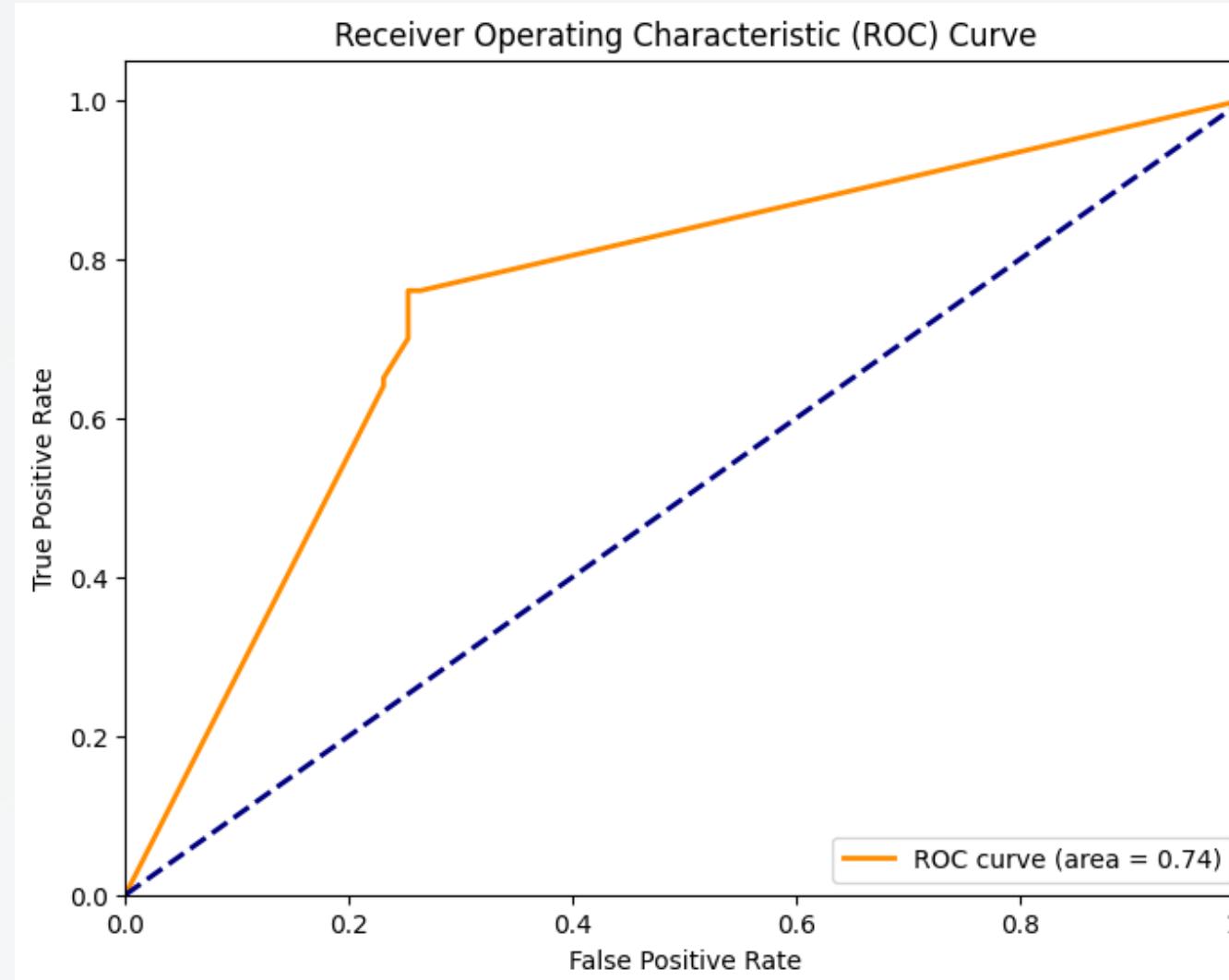
Accuracy score: 0.86
Survivors precision: 0.84
Survivors recall: 0.87
Deceased precision: 0.88
Deceased recall: 0.85

The model exhibits strong performance with an accuracy score of 0.86. Both classes' precision and recall scores are well-balanced, indicating a relatively low degree of bias, and the model effectively captures true positives while minimizing false negatives. The general balance between precision and recall suggests a balanced level of variance. Based on some tests in kaggle, we would say its overfitted.

MÉTRICAS DE EVALUACIÓN -> RANDOM FOREST LDA

{'bootstrap': False, 'max_depth': 20, 'n_estimators': 100, 'random_state': 42}

Best aggregate performance: 0.78



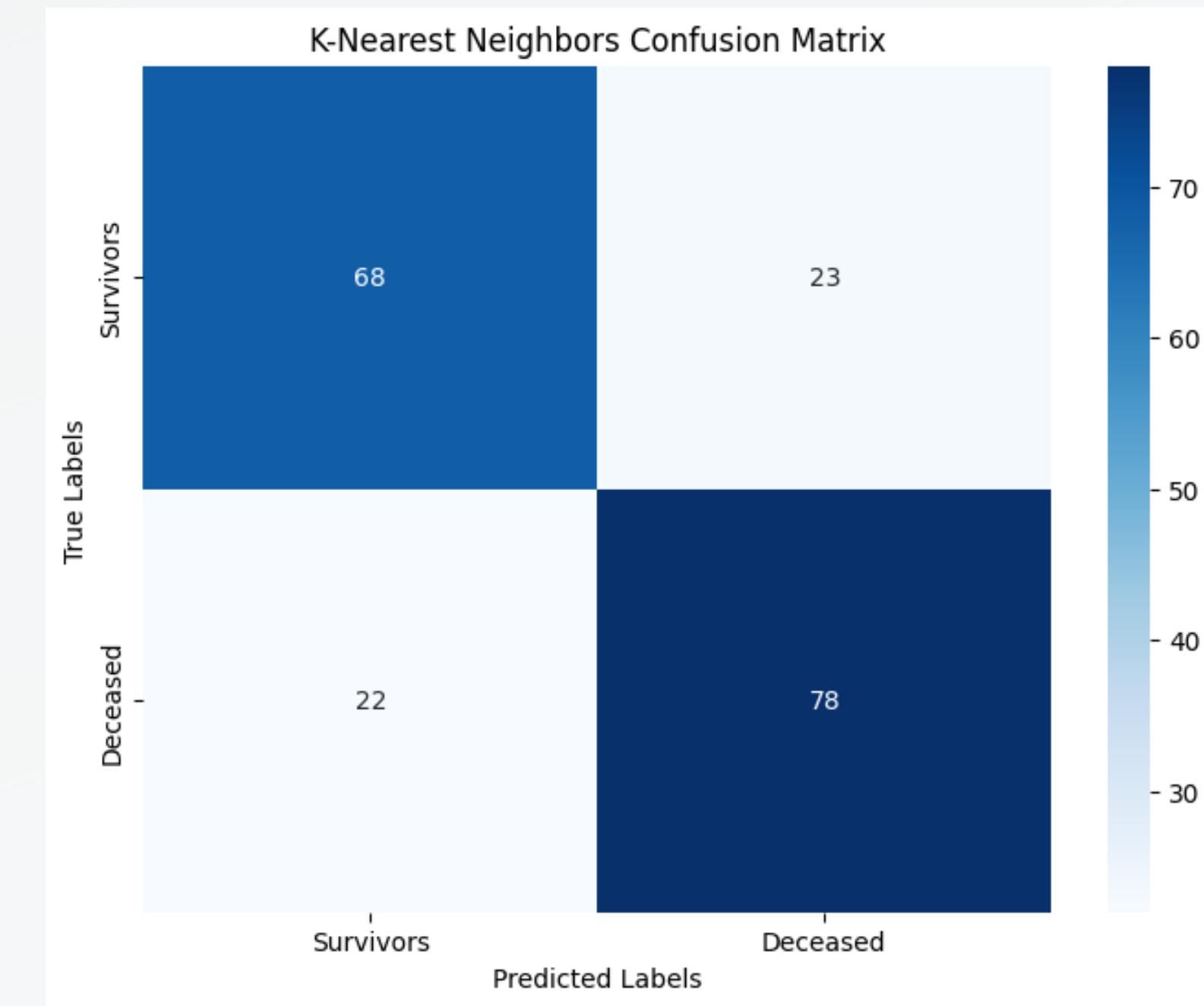
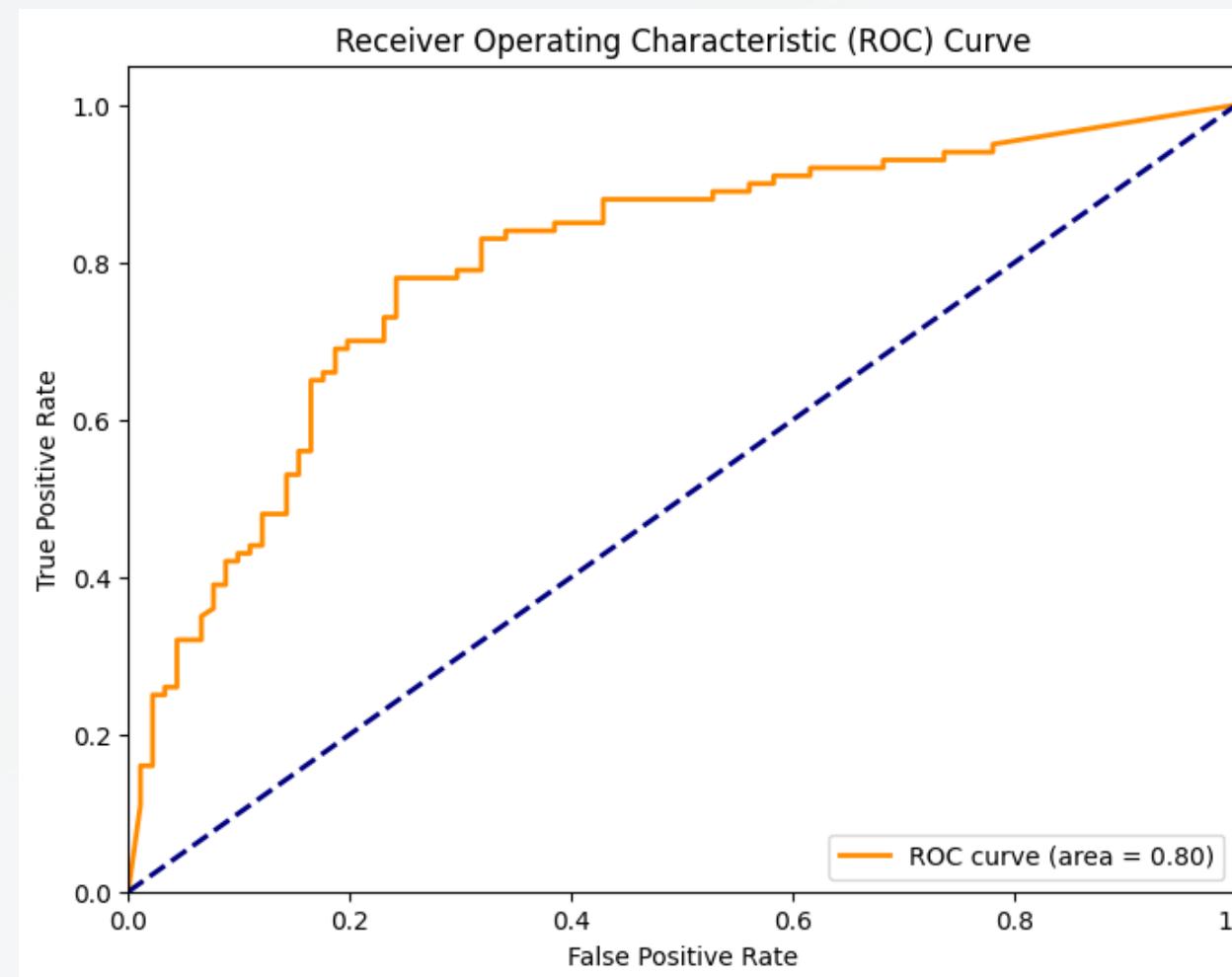
Accuracy score: 0.71 Survivors
precision: 0.67
Survivors recall: 0.77 Deceased
precision: 0.76
Deceased recall: 0.65

In the Accuracy score of Random Forest with our LDA were higher from the one with LDA, this is because random forest need features in order have a better classification, the recall indicate a good balance between both classes, indicating that this is a good classification also thanks that Random Forest is a good method to deal with overfitting and large amount of features that is the case in this dataset. The confusion matrix exhibits that this model has a bias towards deceased people as survivors.

MÉTRICAS DE EVALUACIÓN -> KNNNEIGHBORS

{'n_neighbors': 7, 'weights': 'distance'}

Best aggregate performance: 0.72

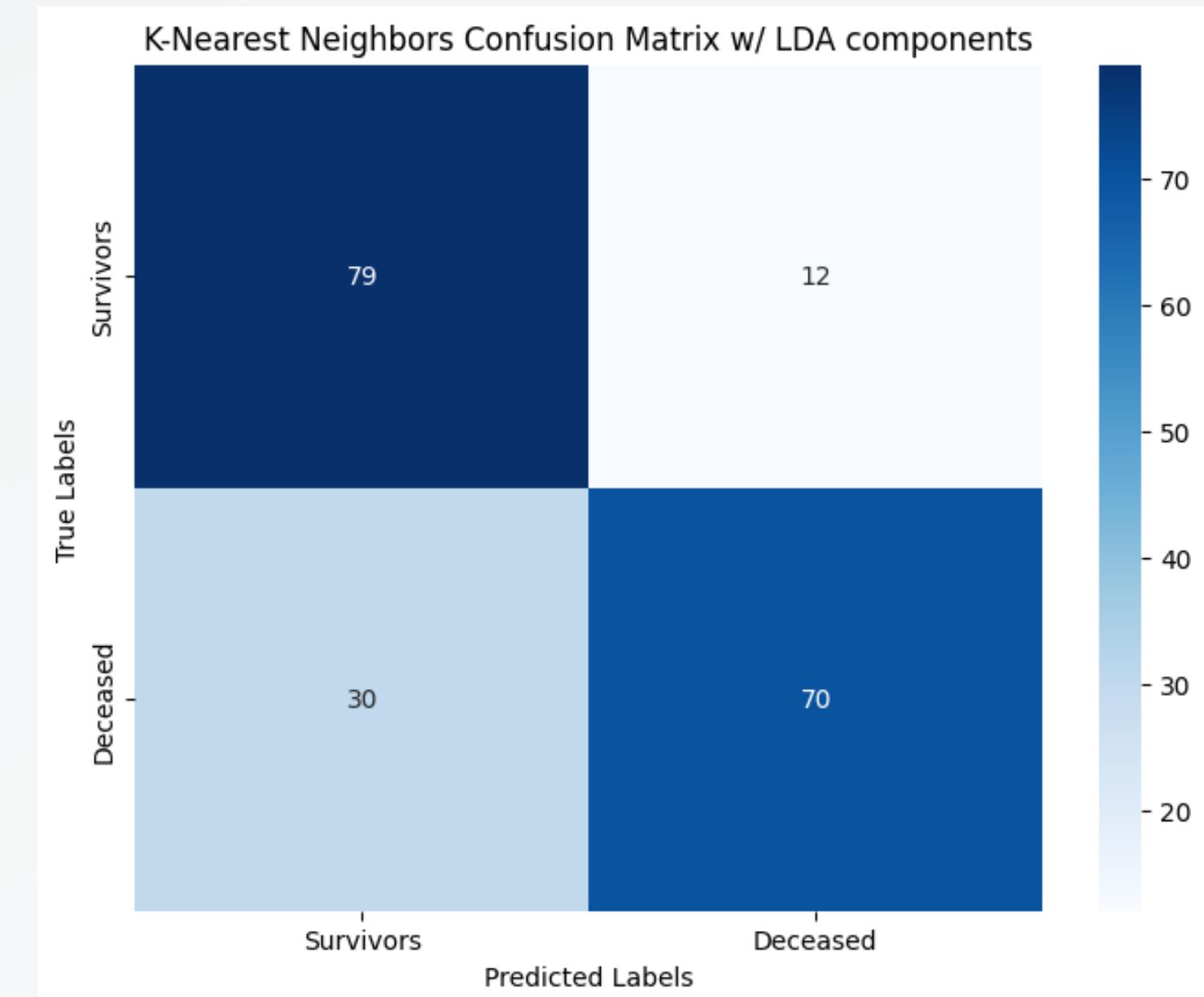
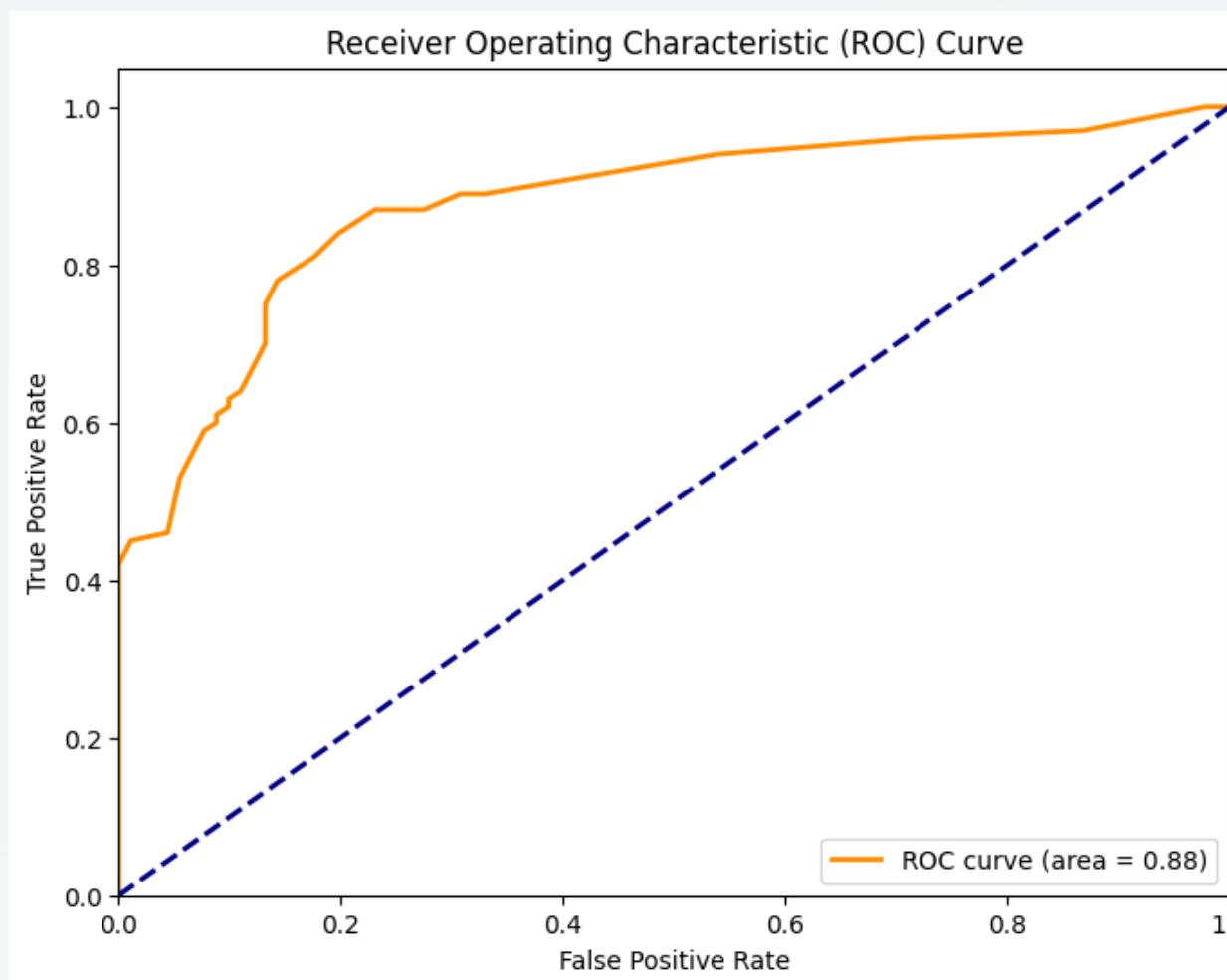


The accuracy score of 0.76 is somewhat accurate classification. Survivors and deceased's precision and recall scores are consistent and balanced, suggesting a moderate level of bias with the model. The relatively lower precision and recall scores compared to the accuracy score indicate a moderate level of variance, meaning there is room for improvement in capturing true positives and reducing false negatives. Overall, the model exhibits a balance between bias and variance.

MÉTRICAS DE EVALUACIÓN -> KKNEIGHBORS LDA

{'n_neighbors': 27, 'weights': 'uniform'}

Best aggregate performance: 0.82



Accuracy score: 0.78
Survivors precision: 0.72
Survivors recall: 0.87
Deceased precision: 0.85
Deceased recall: 0.70

Precision and recall scores for survivors and deceased are similar and well-balanced, indicating a moderate level of bias in the model. The somewhat lower precision and recall scores in comparison to accuracy indicate a moderate level of variance, implying there is room for improvement in correctly identifying true positives and reducing false negatives.

MÉTRICAS DE EVALUACIÓN -> BYNARY FEEDFORWARD NETWORK (PYTORCH)

```
criterion = nn.BCELoss() # Binary cross entropy
```

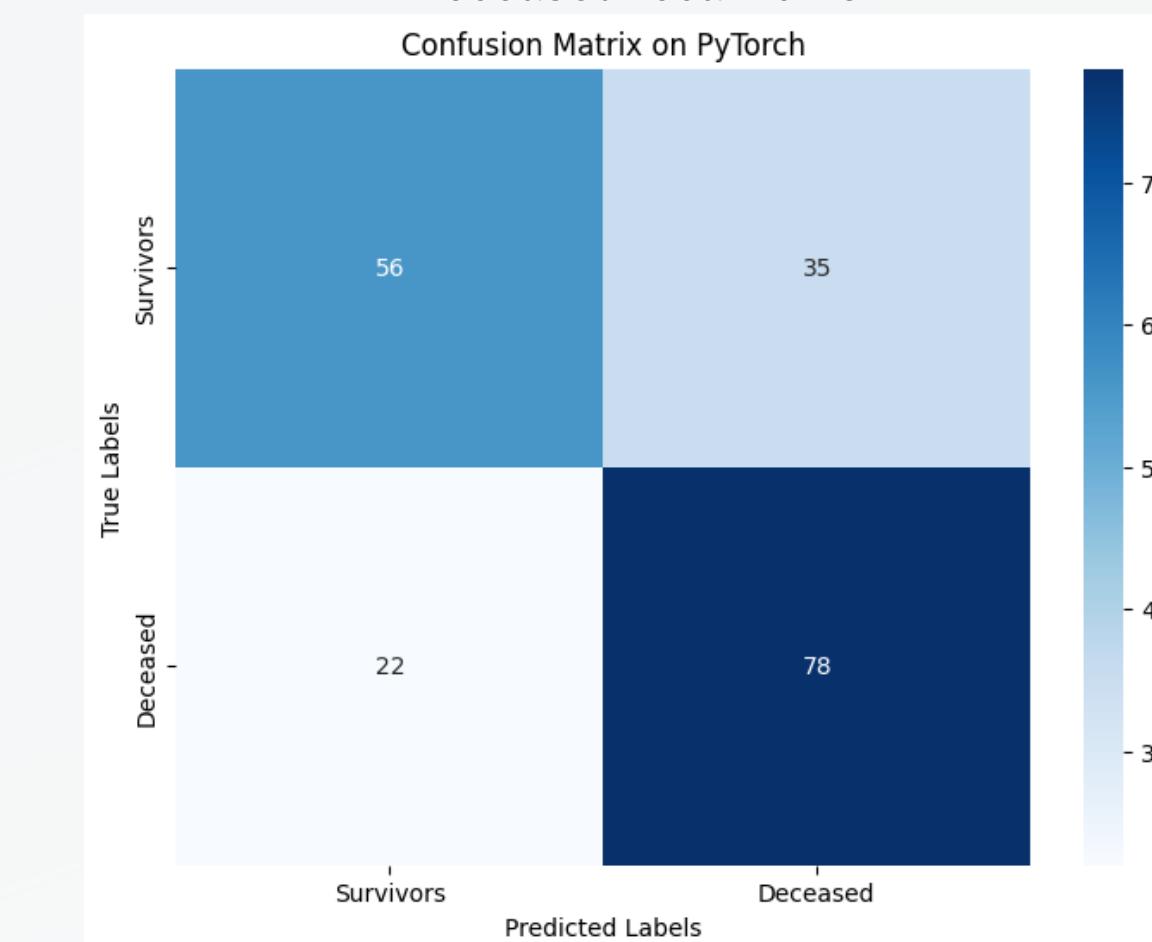
```
model = ClassifierNetwork(X_train_local.shape[1], 128, 1)
```

```
optimizer = torch.optim.Adam(model.parameters(), lr=0.0001)
```

```
train_dataset = TensorDataset(X_train_model, y_train_model)
```

```
batch_size = 32
```

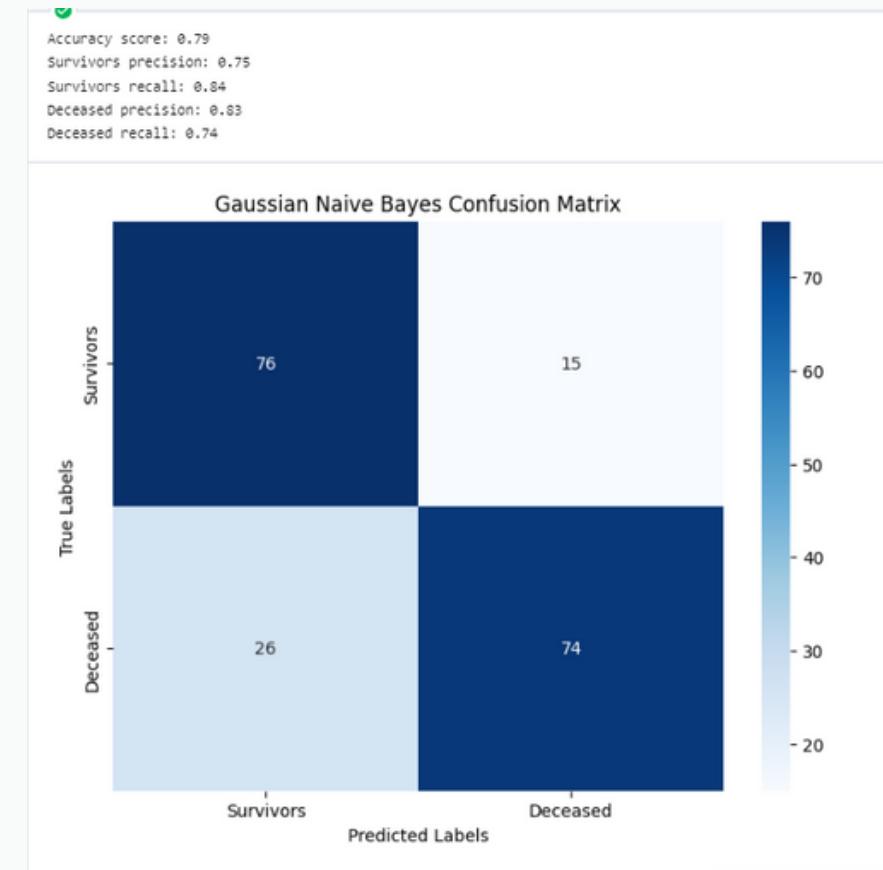
Accuracy score: 0.70
Survivors precision: 0.72
Survivors recall: 0.62
Deceased precision: 0.69
Deceased recall: 0.78



The results from our PyTorch model don't show promising performance with a test accuracy of 62.25% and a test loss of 0.6822. When we evaluated the predictions using a confusion matrix, we found that our model achieved balanced precision and recall scores of 0.62 for both survivors and deceased passengers. This indicates that our PyTorch model is effectively classifying passengers' survival status with equal precision and recall, contributing to an overall robust performance.

EL MEJOR

Gaussian NB w/ LDA Components



gnb_lda_submission.csv
Complete · now
0.7799