LABORATORIO 8

Marianna Flores

20180040

PARTE 1

In [1]:
```
import numpy as np
import pandas as pd
from scipy import stats
from sklearn.impute import SimpleImputer
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.linear_model import LinearRegression
from sklearn.preprocessing import StandardScaler
from sklearn.preprocessing import MinMaxScaler
from sklearn.preprocessing import MaxAbsScaler
```

In [2]:
```
df = pd.read_csv('titanic.csv')
df1 = pd.read_csv('titanic_MD.csv')
```

In [3]:
```
print('titanicmd dataset shape:', df1.shape)
df1.head()
```

```
titanicmd dataset shape: (183, 12)
```

Out[3]:

	PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare	Cabin	Embarked
0	2	1	1	Cumings, Mrs. John Bradley (Florence Briggs Th...	?	38.0	1.0	0.0	PC 17599	71.2833	C85	C
1	4	1	1	Futrelle, Mrs. Jacques Heath (Lily May Peel)	female	35.0	1.0	0.0	113803	53.1000	C123	S
2	7	0	1	McCarthy, Mr. Timothy J	male	54.0	0.0	0.0	17463	51.8625	E46	S
3	11	1	3	Sandstrom, Miss. Marguerite Rut	female	NaN	1.0	NaN	PP 9549	16.7000	G6	S
4	12	1	1	Bonnell, Miss. Elizabeth	female	58.0	NaN	0.0	113783	26.5500	C103	S

In [4]:
```
df1.describe()
```

Out[4]:

	PassengerId	Survived	Pclass	Age	SibSp	Parch	Fare
count	183.000000	183.000000	183.000000	158.000000	180.000000	171.000000	175.000000
mean	455.366120	0.672131	1.191257	35.692532	0.461111	0.461988	78.959191
std	247.052476	0.470725	0.515187	15.640858	0.646122	0.753435	77.026328
min	2.000000	0.000000	1.000000	0.920000	0.000000	0.000000	0.000000
25%	263.500000	0.000000	1.000000	24.000000	0.000000	0.000000	29.700000
50%	457.000000	1.000000	1.000000	35.500000	0.000000	0.000000	56.929200
75%	676.000000	1.000000	1.000000	48.000000	1.000000	1.000000	90.539600
max	890.000000	1.000000	3.000000	80.000000	3.000000	4.000000	512.329200

In [5]:
```
df1.isna().sum()
```

Out[5]:
```
PassengerId     0
Survived        0
Pclass          0
Name            0
Sex             0
Age            25
SibSp           3
Parch          12
Ticket          0
Fare            8
Cabin           0
Embarked       12
dtype: int64
```

In [6]:
```
cols = []
val = []
for col in df1.select_dtypes(include='object').columns:
    cols.append(col)
    val.append(df1[col].str.contains(r'\?').sum())
pd.DataFrame({
    'cols':cols,
    'val':val
})
```

Out[6]:

	cols	val
0	Name	0
1	Sex	51
2	Ticket	0
3	Cabin	0
4	Embarked	0

In [7]:
```
df1.replace(r'\?', np.nan, regex = True, inplace = True)
df1.isna().sum()
```

Out[7]:
```
PassengerId     0
Survived        0
Pclass          0
Name            0
Sex            51
Age            25
SibSp           3
Parch          12
Ticket          0
Fare            8
Cabin           0
Embarked       12
dtype: int64
```

1. MISSING DATA POR COLUMNAS

Las columnas con data faltante son las siguientes:
Sex - 51, forma de faltantes: "?" (categorico)
Age - 25, faltantes NaN
SibSp - 3, faltantes NaN
Parch - 12, faltantes NaN
Fare - 8, faltantes NaN
Embarked - 12, faltantes NaN (categorico)

2. MODELOS Y VALORES

Sex - modelo de imputacion general con moda, por ser datos categoricos se reemplaza con el mas repetido.
Age - modelo Regresion Lineal, segun el comportamiento de los datos cual es el dato de edad mas probable y atinado.
SibSp - modelo Pairwise deletion, por ser pocos datos no el efecto de eliminar los nas no es tan notorio y puede ser efectivo para el caso de esta variable.
Parch - modelo outliers percentil, va a distribuir los datos segun las cualidades maximas y minimas de los datos.
Fare - modelo outliers standard deviation, los datos de la tarifa son los mas variables y complejo por lo que un modelo predictivo podria ser mejor para encontrar datos mas precisos segun el comportamiento.
Embarked - modelo de imputacion general con moda, por ser datos categoricos se reemplaza con el mas repetido.

3. FILAS COMPLETAS

Las filas que no cuentan con NAs son las siguientes:
PassengerId
Survived
Pclass
Name
Ticket
Cabin

4. METODOS MISSING VALUES

a. Pairwise Deletion

In [8]:
```
df1.corr()
```

Out[8]:

	PassengerId	Survived	Pclass	Age	SibSp	Parch	Fare
PassengerId	1.000000	0.148495	-0.089136	-0.048190	-0.088806	-0.062083	0.022261
Survived	0.148495	1.000000	-0.034542	-0.257703	0.113987	-0.003365	0.119311
Pclass	-0.089136	-0.034542	1.000000	-0.297872	-0.102294	0.041969	-0.304438
Age	-0.048190	-0.257703	-0.297872	1.000000	-0.087951	-0.279548	-0.130979
SibSp	-0.088806	0.113987	-0.102294	-0.087951	1.000000	0.255152	0.299061
Parch	-0.062083	-0.003365	0.041969	-0.279548	0.255152	1.000000	0.381445
Fare	0.022261	0.119311	-0.304438	-0.130979	0.299061	0.381445	1.000000

b. Imputacion General (media, mediana, moda)

In [9]:
```
imp_mean = SimpleImputer(missing_values=np.nan, strategy='mean')
imp_median = SimpleImputer(missing_values=np.nan, strategy='median')
imp_mode = SimpleImputer(missing_values=np.nan, strategy='most_frequent')
```

In [10]:
```
## media
p4b1= df1
p4b1[['Age']] = imp_mean.fit_transform(p4b1[['Age']])
p4b1[['SibSp']] = imp_mean.fit_transform(p4b1[['SibSp']])
p4b1[['Parch']] = imp_mean.fit_transform(p4b1[['Parch']])
p4b1[['Fare']] = imp_mean.fit_transform(p4b1[['Fare']])
```

In [11]:
```
## mediana
p4b2= df1
p4b2[['Age']] = imp_median.fit_transform(p4b2[['Age']])
p4b2[['SibSp']] = imp_median.fit_transform(p4b2[['SibSp']])
p4b2[['Parch']] = imp_median.fit_transform(p4b2[['Parch']])
p4b2[['Fare']] = imp_median.fit_transform(p4b2[['Fare']])
```

In [12]:
```
## moda
p4b3= df1
p4b3[['Sex']] = imp_mode.fit_transform(p4b3[['Sex']])
p4b3[['Age']] = imp_mode.fit_transform(p4b3[['Age']])
p4b3[['SibSp']] = imp_mode.fit_transform(p4b3[['SibSp']])
p4b3[['Parch']] = imp_mode.fit_transform(p4b3[['Parch']])
p4b3[['Fare']] = imp_mode.fit_transform(p4b3[['Fare']])
p4b3[['Embarked']] = imp_mode.fit_transform(p4b3[['Embarked']])
```

c. Imputacion sectorizada

In [14]:
```
diccionario1 = pd.cut(df1,['Sex'], 2)
diccionario2 = pd.cut(df1[['Age']], 5)
diccionario3 = pd.cut(df1[['SibSp']], 3)
diccionario4 = pd.cut(df1[['Parch']], 3)
diccionario5 = pd.cut(df1[['Fare']], 3)
diccionario6 = pd.cut(df1[['Embarked']], 5)
diccionario = pd.DataFrame(diccionario1, diccionario2, diccionario3, diccionario4, dicci
onario5, diccionario6)
```

In [15]:
```
d4c = df1
d4c = d4c.merge(diccionario, how = 'left')
d4c['Sex2'] = np.where(df['Sex'].isna(), df['Sex_new'], df['Sex'])
d4c['Age2'] = np.where(df['Age'].isna(), df['Age_new'], df['Age'])
d4c['SibSp2'] = np.where(df['SibSp'].isna(), df['SibSp_new'], df['SibSp'])
d4c['Parch2'] = np.where(df['Parch'].isna(), df['Parch_new'], df['Parch'])
d4c['Fare2'] = np.where(df['Fare'].isna(), df['Fare_new'], df['Fare'])
d4c['Embarked2'] = np.where(df['Embarked'].isna(), df['Embarked_new'], df['Embarked'])
```

d. Modelo de Regresion Lineal

In [16]:
```
p4d = df1
lm = LinearRegression()
lm1 = lm.fit(p4d[['Age']], p4d['Age'])
p4d['Age'] = lm1.predict(p4d[['Age']])

lm2 = lm.fit(p4d[['SibSp']], p4d['SibSp'])
p4d['SibSp'] = lm2.predict(p4d[['SibSp']])

lm3 = lm.fit(p4d[['Parch']], p4d['Parch'])
p4d['Parch'] = lm3.predict(p4d[['Parch']])

lm4 = lm.fit(p4d[['Fare']], p4d['Fare'])
p4d['Fare'] = lm4.predict(p4d[['Fare']])
```

e. Outliers: St. dev. approach

In [17]:

```
f = 2
p4e = df1
xl = p4e['Age'].mean() - (p4e['Age'].std() * f)
xu = p4e['Age'].mean() + (p4e['Age'].std() * f)
df_sd = p4e[(p4e['Age']>=xl) & (p4e['Age']<=xu)]
p4e['Age'] = np.where(
    p4e['Age']<xl,
    xl,
    np.where(
        p4e['Age']>xu,
        xu,
        p4e['Age']
    )
)
xl = p4e['SibSp'].mean() - (p4e['SibSp'].std() * f)
xu = p4e['SibSp'].mean() + (p4e['SibSp'].std() * f)
df_sd = p4e[(p4e['SibSp']>=xl) & (p4e['SibSp']<=xu)]
p4e['SibSp'] = np.where(
    p4e['SibSp']<xl,
    xl,
    np.where(
        p4e['SibSp']>xu,
        xu,
        p4e['SibSp']
    )
)
xl = p4e['Parch'].mean() - (p4e['Parch'].std() * f)
xu = p4e['Parch'].mean() + (p4e['Parch'].std() * f)
df_sd = p4e[(p4e['Age']>=xl) & (p4e['Parch']<=xu)]
p4e['Parch'] = np.where(
    p4e['Parch']<xl,
    xl,
    np.where(
        p4e['Parch']>xu,
        xu,
        p4e['Parch']
    )
)
xl = p4e['Fare'].mean() - (p4e['Fare'].std() * f)
xu = p4e['Fare'].mean() + (p4e['Fare'].std() * f)
df_sd = p4e[(p4e['Fare']>=xl) & (p4e['Fare']<=xu)]
p4e['Fare'] = np.where(
    p4e['Fare']<xl,
    xl,
    np.where(
        p4e['Fare']>xu,
        xu,
        p4e['Fare']
    )
)
```

f. Outliers: Percentile approach

In [18]:

```
p4f = df1
pl = np.percentile(p4f['Age'], 5)
pu = np.percentile(p4f['Age'], 95)
df_sd = p4f[(p4f['Age']>=pl) & (p4f['Age']<=pu)]
p4f['Age'] = np.where(
    p4f['Age']<pl,
    pl,
    np.where(
        p4f['Age']>pu,
        pu,
        p4f['Age']
    )
)
pl = np.percentile(p4f['SibSp'], 5)
pu = np.percentile(p4f['SibSp'], 95)
df_sd = p4f[(p4f['SibSp']>=pl) & (p4f['SibSp']<=pu)]
p4f['SibSp'] = np.where(
    p4f['SibSp']<pl,
    pl,
    np.where(
        p4f['SibSp']>pu,
        pu,
        p4f['SibSp']
    )
)
pl = np.percentile(p4f['Parch'], 5)
pu = np.percentile(p4f['Parch'], 95)
df_sd = p4f[(p4f['Parch']>=pl) & (p4f['Parch']<=pu)]
p4f['Parch'] = np.where(
    p4f['Parch']<pl,
    pl,
    np.where(
        p4f['Parch']>pu,
        pu,
        p4f['Parch']
    )
)
pl = np.percentile(p4f['Fare'], 5)
pu = np.percentile(p4f['Fare'], 95)
df_sd = p4f[(p4f['Fare']>=pl) & (p4f['Fare']<=pu)]
p4f['Fare'] = np.where(
    p4f['Fare']<pl,
    pl,
    np.where(
        p4f['Fare']>pu,
        pu,
        p4f['Fare']
    )
)
```

5. COMPARACION

In [19]:
```
base = df[['Sex', 'Age','SibSp','Parch','Fare','Embarked']]
base.head()
```

Out[19]:

	Sex	Age	SibSp	Parch	Fare	Embarked
0	female	38.0	1	0	71.2833	C
1	female	35.0	1	0	53.1000	S
2	male	54.0	0	0	51.8625	S
3	female	4.0	1	1	16.7000	S
4	female	58.0	0	0	26.5500	S

In [20]:
```
pb1 = p4b1[['Sex', 'Age','SibSp','Parch','Fare','Embarked']]
pb1.head()
```

Out[20]:

	Sex	Age	SibSp	Parch	Fare	Embarked
0	male	38.000000	1.000000e+00	-3.330669e-16	71.2833	C
1	female	35.000000	1.000000e+00	-3.330669e-16	53.1000	S
2	male	54.000000	2.220446e-16	-3.330669e-16	51.8625	S
3	female	35.692532	1.000000e+00	4.619883e-01	16.7000	S
4	female	58.000000	4.611111e-01	-3.330669e-16	26.5500	S

In [21]:
```
pb2 = p4b2[['Sex', 'Age','SibSp','Parch','Fare','Embarked']]
pb2.head()
```

Out[21]:

	Sex	Age	SibSp	Parch	Fare	Embarked
0	male	38.000000	1.000000e+00	-3.330669e-16	71.2833	C
1	female	35.000000	1.000000e+00	-3.330669e-16	53.1000	S
2	male	54.000000	2.220446e-16	-3.330669e-16	51.8625	S
3	female	35.692532	1.000000e+00	4.619883e-01	16.7000	S
4	female	58.000000	4.611111e-01	-3.330669e-16	26.5500	S

In [22]:
```
pb3 = p4b3[['Sex', 'Age','SibSp','Parch','Fare','Embarked']]
pb3.head()
```

Out[22]:

	Sex	Age	SibSp	Parch	Fare	Embarked
0	male	38.000000	1.000000e+00	-3.330669e-16	71.2833	C
1	female	35.000000	1.000000e+00	-3.330669e-16	53.1000	S
2	male	54.000000	2.220446e-16	-3.330669e-16	51.8625	S
3	female	35.692532	1.000000e+00	4.619883e-01	16.7000	S
4	female	58.000000	4.611111e-01	-3.330669e-16	26.5500	S

In [23]:
```
pc = p4c[['Sex', 'Age','SibSp','Parch','Fare','Embarked']]
pc.head()
```

Out[23]:

	Sex	Age	SibSp	Parch	Fare	Embarked
0	male	38.000000	1.000000e+00	-3.330669e-16	71.2833	C
1	female	35.000000	1.000000e+00	-3.330669e-16	53.1000	S
2	male	54.000000	2.220446e-16	-3.330669e-16	51.8625	S
3	female	35.692532	1.000000e+00	4.619883e-01	16.7000	S
4	female	58.000000	4.611111e-01	-3.330669e-16	26.5500	S

In [24]:
```
pd = p4d[['Sex', 'Age','SibSp','Parch','Fare','Embarked']]
pd.head()
```

Out[24]:

	Sex	Age	SibSp	Parch	Fare	Embarked
0	male	38.000000	1.000000e+00	-3.330669e-16	71.2833	C
1	female	35.000000	1.000000e+00	-3.330669e-16	53.1000	S
2	male	54.000000	2.220446e-16	-3.330669e-16	51.8625	S
3	female	35.692532	1.000000e+00	4.619883e-01	16.7000	S
4	female	58.000000	4.611111e-01	-3.330669e-16	26.5500	S

In [25]:
```
pe = p4e[['Sex', 'Age','SibSp','Parch','Fare','Embarked']]
pe.head()
```

Out[25]:

	Sex	Age	SibSp	Parch	Fare	Embarked
0	male	38.000000	1.000000e+00	-3.330669e-16	71.2833	C
1	female	35.000000	1.000000e+00	-3.330669e-16	53.1000	S
2	male	54.000000	2.220446e-16	-3.330669e-16	51.8625	S
3	female	35.692532	1.000000e+00	4.619883e-01	16.7000	S
4	female	58.000000	4.611111e-01	-3.330669e-16	26.5500	S

In [26]:
```
pf = p4f[['Sex', 'Age','SibSp','Parch','Fare','Embarked']]
pf.head()
```

Out[26]:

	Sex	Age	SibSp	Parch	Fare	Embarked
0	male	38.000000	1.000000e+00	-3.330669e-16	71.2833	C
1	female	35.000000	1.000000e+00	-3.330669e-16	53.1000	S
2	male	54.000000	2.220446e-16	-3.330669e-16	51.8625	S
3	female	35.692532	1.000000e+00	4.619883e-01	16.7000	S
4	female	58.000000	4.611111e-01	-3.330669e-16	26.5500	S

Los modelos recomendados para cada columna con distintos nas son los siguientes:
Sex - Imputacion General Moda
Age - Imputacion general Media
SibSp - Eliminacion Outliers: SDA
Parch - Eliminacion Outliers: PA
Fare - Modelo de regresion lineal simple
Embarked - Imputacion Sectorizada
Estos modelos son los mas adecuados para cada una de estas columnas dado que los modelos se adaptaron mas al comportamiento de los datos y caracteristicas de estos. Todos los modelos dieron resultados muy similares por lo que la mejor forma de definir el mas adecuado era por medio del tipo de datos que se estaba manejando y como cada modelo se adaptaba a las necesidades de estos.

6. CONCLUSIONES

Se obtuvieron tres principales conclusiones de este proceso:

1. Tipo de dato: Cada tipo de dato se tiene que entender inicialmente para poder aplicarle los/el modelo mas conveniente y eficiente.
2. Variedad de modelos: Es importante analizar varios tipos de modelos para estar seguro de estar obteniendo resultados reales y confiables.
3. Modelos predictivos: Este tipo de modelo aunque es mas complejo tiene muchisimo potencial para dar los resultados mas confiables y basados en el comportamiento de los datos.

PARTE 2

1. NORMALIZACION COLUMNAS

a. Standarization

In [27]:
```
scaler1 = StandardScaler()
df_1 = df.copy()
for col in df_1.select_dtypes(include=['float', 'int']).columns:
    df_1[col+'_1'] = scaler1.fit_transform(df_1[[col]])

pb1_1 = pb1.copy()
for col in pb1_1.select_dtypes(include=['float', 'int']).columns:
    pb1_1[col+'_1'] = scaler1.fit_transform(pb1_1[[col]])

pb2_1 = pb2.copy()
for col in pb2_1.select_dtypes(include=['float', 'int']).columns:
    pb2_1[col+'_1'] = scaler1.fit_transform(pb2_1[[col]])

pb3_1 = pb3.copy()
for col in pb3_1.select_dtypes(include=['float', 'int']).columns:
    pb3_1[col+'_1'] = scaler1.fit_transform(pb3_1[[col]])

pc_1 = pc.copy()
for col in pc_1.select_dtypes(include=['float', 'int']).columns:
    pc_1[col+'_1'] = scaler1.fit_transform(pc_1[[col]])

pd_1 = pd.copy()
for col in pd_1.select_dtypes(include=['float', 'int']).columns:
    pd_1[col+'_1'] = scaler1.fit_transform(pd_1[[col]])

pe_1 = pe.copy()
for col in pe_1.select_dtypes(include=['float', 'int']).columns:
    pe_1[col+'_1'] = scaler1.fit_transform(pe_1[[col]])

pf_1 = pf.copy()
for col in pf_1.select_dtypes(include=['float', 'int']).columns:
    pf_1[col+'_1'] = scaler1.fit_transform(pf_1[[col]])
```

b. MinMaxScaling

In [28]:
```
scaler2 = MinMaxScaler()
df_2 = df.copy()
for col in df_2.select_dtypes(include=['float', 'int']).columns:
    df_2[col+'_2'] = scaler2.fit_transform(df_2[[col]])

pb1_2 = pb1.copy()
for col in pb1_2.select_dtypes(include=['float', 'int']).columns:
    pb1_2[col+'_2'] = scaler2.fit_transform(pb1_2[[col]])

pb2_2 = pb2.copy()
for col in pb2_2.select_dtypes(include=['float', 'int']).columns:
    pb2_2[col+'_2'] = scaler2.fit_transform(pb2_2[[col]])

pb3_2 = pb3.copy()
for col in pb3_2.select_dtypes(include=['float', 'int']).columns:
    pb3_2[col+'_2'] = scaler2.fit_transform(pb3_2[[col]])

pc_2 = pc.copy()
for col in pc_2.select_dtypes(include=['float', 'int']).columns:
    pc_2[col+'_2'] = scaler2.fit_transform(pc_2[[col]])

pd_2 = pd.copy()
for col in pd_2.select_dtypes(include=['float', 'int']).columns:
    pd_2[col+'_2'] = scaler2.fit_transform(pd_2[[col]])

pe_2 = pe.copy()
for col in pe_2.select_dtypes(include=['float', 'int']).columns:
    pe_2[col+'_2'] = scaler2.fit_transform(pe_2[[col]])

pf_2 = pf.copy()
for col in pf_2.select_dtypes(include=['float', 'int']).columns:
    pf_2[col+'_2'] = scaler2.fit_transform(pf_2[[col]])
```

c. MaxAbsScaler

In [29]:
```
scaler3 = MaxAbsScaler()
df_3 = df.copy()
for col in df_3.select_dtypes(include=['float', 'int']).columns:
    df_3[col+'_3'] = scaler3.fit_transform(df_3[[col]])

pb1_3 = pb1.copy()
for col in pb1_3.select_dtypes(include=['float', 'int']).columns:
    pb1_3[col+'_3'] = scaler3.fit_transform(pb1_3[[col]])

pb2_3 = pb2.copy()
for col in pb2_3.select_dtypes(include=['float', 'int']).columns:
    pb2_3[col+'_3'] = scaler3.fit_transform(pb2_3[[col]])

pb3_3 = pb3.copy()
for col in pb3_3.select_dtypes(include=['float', 'int']).columns:
    pb3_3[col+'_3'] = scaler3.fit_transform(pb3_3[[col]])

pc_3 = pc.copy()
for col in pc_3.select_dtypes(include=['float', 'int']).columns:
    pc_3[col+'_3'] = scaler3.fit_transform(pc_3[[col]])

pd_3 = pd.copy()
for col in pd_3.select_dtypes(include=['float', 'int']).columns:
    pd_3[col+'_3'] = scaler3.fit_transform(pd_3[[col]])

pe_3 = pe.copy()
for col in pe_3.select_dtypes(include=['float', 'int']).columns:
    pe_3[col+'_3'] = scaler3.fit_transform(pe_3[[col]])

pf_3 = pf.copy()
for col in pf_3.select_dtypes(include=['float', 'int']).columns:
    pf_3[col+'_3'] = scaler3.fit_transform(pf_3[[col]])
```

2. COMPARACION Y CONCLUSION

In [30]:
```
df_1.describe()
```

Out[30]:

	PassengerId	Survived	Pclass	Age	SibSp	Parch	Fare	Age_1
count	183.000000	183.000000	183.000000	183.000000	183.000000	183.000000	183.000000	1.830000e+02
mean	455.366120	0.672131	1.191257	35.674426	0.464481	0.475410	78.682469	-1.650168e-16
std	247.052476	0.470725	0.515187	15.643866	0.644159	0.754617	76.347843	1.002743e+00
min	2.000000	0.000000	1.000000	0.920000	0.000000	0.000000	0.000000	-2.227696e+00
25%	263.500000	0.000000	1.000000	24.000000	0.000000	0.000000	29.700000	-7.483096e-01
50%	457.000000	1.000000	1.000000	36.000000	0.000000	0.000000	57.000000	2.086869e-02
75%	676.000000	1.000000	1.000000	47.500000	1.000000	1.000000	90.000000	7.579979e-01
max	890.000000	1.000000	3.000000	80.000000	3.000000	4.000000	512.329200	2.841189e+00

In [31]:
```
pb1_1.describe()
```

Out[31]:

	Age	SibSp	Parch	Fare	Age_1	SibSp_1	Parch_1	
count	183.000000	1.830000e+02	1.830000e+02	183.000000	1.830000e+02	1.830000e+02	1.830000e+02	1.83(
mean	35.889690	3.955373e-01	4.412416e-01	74.762974	3.300335e-16	2.523786e-16	-8.736181e-17	1.45
std	12.929136	4.861124e-01	6.666807e-01	58.137731	1.002743e+00	1.002743e+00	1.002743e+00	1.00:
min	15.100000	2.220446e-16	-3.330669e-16	10.500000	-1.612384e+00	-8.159070e-01	-6.636642e-01	-1.10(
25%	25.000000	2.220446e-16	-3.330669e-16	30.000000	-8.445704e-01	-8.159070e-01	-6.636642e-01	-7.72
50%	35.692532	2.220446e-16	-3.330669e-16	61.175000	-1.529100e-02	-8.159070e-01	-6.636642e-01	-2.34
75%	46.500000	1.000000e+00	1.000000e+00	90.000000	8.229025e-01	1.246874e+00	8.404193e-01	2.62
max	60.900000	1.000000e+00	1.918334e+00	229.381725	1.939722e+00	1.246874e+00	2.221671e+00	2.66(

In [32]:
```
pb2_1.describe()
```

Out[32]:

	Age	SibSp	Parch	Fare	Age_1	SibSp_1	Parch_1	
count	183.000000	1.830000e+02	1.830000e+02	183.000000	1.830000e+02	1.830000e+02	1.830000e+02	1.83(
mean	35.889690	3.955373e-01	4.412416e-01	74.762974	3.300335e-16	2.523786e-16	-8.736181e-17	1.45
std	12.929136	4.861124e-01	6.666807e-01	58.137731	1.002743e+00	1.002743e+00	1.002743e+00	1.00:
min	15.100000	2.220446e-16	-3.330669e-16	10.500000	-1.612384e+00	-8.159070e-01	-6.636642e-01	-1.10(
25%	25.000000	2.220446e-16	-3.330669e-16	30.000000	-8.445704e-01	-8.159070e-01	-6.636642e-01	-7.72
50%	35.692532	2.220446e-16	-3.330669e-16	61.175000	-1.529100e-02	-8.159070e-01	-6.636642e-01	-2.34
75%	46.500000	1.000000e+00	1.000000e+00	90.000000	8.229025e-01	1.246874e+00	8.404193e-01	2.62
max	60.900000	1.000000e+00	1.918334e+00	229.381725	1.939722e+00	1.246874e+00	2.221671e+00	2.66(

In [33]:
```
pb3_1.describe()
```

Out[33]:

	Age	SibSp	Parch	Fare	Age_1	SibSp_1	Parch_1	
count	183.000000	1.830000e+02	1.830000e+02	183.000000	1.830000e+02	1.830000e+02	1.830000e+02	1.83(
mean	35.889690	3.955373e-01	4.412416e-01	74.762974	3.300335e-16	2.523786e-16	-8.736181e-17	1.45
std	12.929136	4.861124e-01	6.666807e-01	58.137731	1.002743e+00	1.002743e+00	1.002743e+00	1.00
min	15.100000	2.220446e-16	-3.330669e-16	10.500000	-1.612384e+00	-8.159070e-01	-6.636642e-01	-1.10
25%	25.000000	2.220446e-16	-3.330669e-16	30.000000	-8.445704e-01	-8.159070e-01	-6.636642e-01	-7.72
50%	35.692532	2.220446e-16	-3.330669e-16	61.175000	-1.529100e-02	-8.159070e-01	-6.636642e-01	-2.34
75%	46.500000	1.000000e+00	1.000000e+00	90.000000	8.229025e-01	1.246874e+00	8.404193e-01	2.62
max	60.900000	1.000000e+00	1.918334e+00	229.381725	1.939722e+00	1.246874e+00	2.221671e+00	2.66(

In [47]:
```
pc_1.describe()
```

Out[47]:

	Age	SibSp	Parch	Fare	Age_1	SibSp_1	Parch_1	
count	183.000000	1.830000e+02	1.830000e+02	183.000000	1.830000e+02	1.830000e+02	1.830000e+02	1.83(
mean	35.889690	3.955373e-01	4.412416e-01	74.762974	3.300335e-16	2.523786e-16	-8.736181e-17	1.45
std	12.929136	4.861124e-01	6.666807e-01	58.137731	1.002743e+00	1.002743e+00	1.002743e+00	1.00
min	15.100000	2.220446e-16	-3.330669e-16	10.500000	-1.612384e+00	-8.159070e-01	-6.636642e-01	-1.10
25%	25.000000	2.220446e-16	-3.330669e-16	30.000000	-8.445704e-01	-8.159070e-01	-6.636642e-01	-7.72
50%	35.692532	2.220446e-16	-3.330669e-16	61.175000	-1.529100e-02	-8.159070e-01	-6.636642e-01	-2.34
75%	46.500000	1.000000e+00	1.000000e+00	90.000000	8.229025e-01	1.246874e+00	8.404193e-01	2.62
max	60.900000	1.000000e+00	1.918334e+00	229.381725	1.939722e+00	1.246874e+00	2.221671e+00	2.66(

In [35]:
```
pd_1.describe()
```

Out[35]:

	Age	SibSp	Parch	Fare	Age_1	SibSp_1	Parch_1	
count	183.000000	1.830000e+02	1.830000e+02	183.000000	1.830000e+02	1.830000e+02	1.830000e+02	1.83(
mean	35.889690	3.955373e-01	4.412416e-01	74.762974	3.300335e-16	2.523786e-16	-8.736181e-17	1.45
std	12.929136	4.861124e-01	6.666807e-01	58.137731	1.002743e+00	1.002743e+00	1.002743e+00	1.00:
min	15.100000	2.220446e-16	-3.330669e-16	10.500000	-1.612384e+00	-8.159070e-01	-6.636642e-01	-1.10(
25%	25.000000	2.220446e-16	-3.330669e-16	30.000000	-8.445704e-01	-8.159070e-01	-6.636642e-01	-7.72
50%	35.692532	2.220446e-16	-3.330669e-16	61.175000	-1.529100e-02	-8.159070e-01	-6.636642e-01	-2.34
75%	46.500000	1.000000e+00	1.000000e+00	90.000000	8.229025e-01	1.246874e+00	8.404193e-01	2.62
max	60.900000	1.000000e+00	1.918334e+00	229.381725	1.939722e+00	1.246874e+00	2.221671e+00	2.66(

In [36]:
```
pe_1.describe()
```

Out[36]:

	Age	SibSp	Parch	Fare	Age_1	SibSp_1	Parch_1	
count	183.000000	1.830000e+02	1.830000e+02	183.000000	1.830000e+02	1.830000e+02	1.830000e+02	1.83(
mean	35.889690	3.955373e-01	4.412416e-01	74.762974	3.300335e-16	2.523786e-16	-8.736181e-17	1.45
std	12.929136	4.861124e-01	6.666807e-01	58.137731	1.002743e+00	1.002743e+00	1.002743e+00	1.00:
min	15.100000	2.220446e-16	-3.330669e-16	10.500000	-1.612384e+00	-8.159070e-01	-6.636642e-01	-1.10(
25%	25.000000	2.220446e-16	-3.330669e-16	30.000000	-8.445704e-01	-8.159070e-01	-6.636642e-01	-7.72
50%	35.692532	2.220446e-16	-3.330669e-16	61.175000	-1.529100e-02	-8.159070e-01	-6.636642e-01	-2.34
75%	46.500000	1.000000e+00	1.000000e+00	90.000000	8.229025e-01	1.246874e+00	8.404193e-01	2.62
max	60.900000	1.000000e+00	1.918334e+00	229.381725	1.939722e+00	1.246874e+00	2.221671e+00	2.66(

In [37]:
```
pf_1.describe()
```

Out[37]:

	Age	SibSp	Parch	Fare	Age_1	SibSp_1	Parch_1	
count	183.000000	1.830000e+02	1.830000e+02	183.000000	1.830000e+02	1.830000e+02	1.830000e+02	1.83(
mean	35.889690	3.955373e-01	4.412416e-01	74.762974	3.300335e-16	2.523786e-16	-8.736181e-17	1.45
std	12.929136	4.861124e-01	6.666807e-01	58.137731	1.002743e+00	1.002743e+00	1.002743e+00	1.00:
min	15.100000	2.220446e-16	-3.330669e-16	10.500000	-1.612384e+00	-8.159070e-01	-6.636642e-01	-1.10(
25%	25.000000	2.220446e-16	-3.330669e-16	30.000000	-8.445704e-01	-8.159070e-01	-6.636642e-01	-7.72
50%	35.692532	2.220446e-16	-3.330669e-16	61.175000	-1.529100e-02	-8.159070e-01	-6.636642e-01	-2.34
75%	46.500000	1.000000e+00	1.000000e+00	90.000000	8.229025e-01	1.246874e+00	8.404193e-01	2.62
max	60.900000	1.000000e+00	1.918334e+00	229.381725	1.939722e+00	1.246874e+00	2.221671e+00	2.66(

In [38]:
```
df_2.describe()
```

Out[38]:

	PassengerId	Survived	Pclass	Age	SibSp	Parch	Fare	Age_2	
count	183.000000	183.000000	183.000000	183.000000	183.000000	183.000000	183.000000	183.000000	183
mean	455.366120	0.672131	1.191257	35.674426	0.464481	0.475410	78.682469	0.439484	0
std	247.052476	0.470725	0.515187	15.643866	0.644159	0.754617	76.347843	0.197823	0
min	2.000000	0.000000	1.000000	0.920000	0.000000	0.000000	0.000000	0.000000	0
25%	263.500000	0.000000	1.000000	24.000000	0.000000	0.000000	29.700000	0.291856	0
50%	457.000000	1.000000	1.000000	36.000000	0.000000	0.000000	57.000000	0.443601	(
75%	676.000000	1.000000	1.000000	47.500000	1.000000	1.000000	90.000000	0.589024	0
max	890.000000	1.000000	3.000000	80.000000	3.000000	4.000000	512.329200	1.000000	1

In [39]:
```
pb1_2.describe()
```

Out[39]:

	Age	SibSp	Parch	Fare	Age_2	SibSp_2	Parch_2	Fare_2
count	183.000000	1.830000e+02	1.830000e+02	183.000000	183.000000	183.000000	183.000000	183.000000
mean	35.889690	3.955373e-01	4.412416e-01	74.762974	0.453923	0.395537	0.230013	0.293597
std	12.929136	4.861124e-01	6.666807e-01	58.137731	0.282296	0.486112	0.347531	0.265613
min	15.100000	2.220446e-16	-3.330669e-16	10.500000	0.000000	0.000000	0.000000	0.000000
25%	25.000000	2.220446e-16	-3.330669e-16	30.000000	0.216157	0.000000	0.000000	0.089089
50%	35.692532	2.220446e-16	-3.330669e-16	61.175000	0.449619	0.000000	0.000000	0.231518
75%	46.500000	1.000000e+00	1.000000e+00	90.000000	0.685590	1.000000	0.521286	0.363210
max	60.900000	1.000000e+00	1.918334e+00	229.381725	1.000000	1.000000	1.000000	1.000000

In [40]:
```
pb2_2.describe()
```

Out[40]:

	Age	SibSp	Parch	Fare	Age_2	SibSp_2	Parch_2	Fare_2
count	183.000000	1.830000e+02	1.830000e+02	183.000000	183.000000	183.000000	183.000000	183.000000
mean	35.889690	3.955373e-01	4.412416e-01	74.762974	0.453923	0.395537	0.230013	0.293597
std	12.929136	4.861124e-01	6.666807e-01	58.137731	0.282296	0.486112	0.347531	0.265613
min	15.100000	2.220446e-16	-3.330669e-16	10.500000	0.000000	0.000000	0.000000	0.000000
25%	25.000000	2.220446e-16	-3.330669e-16	30.000000	0.216157	0.000000	0.000000	0.089089
50%	35.692532	2.220446e-16	-3.330669e-16	61.175000	0.449619	0.000000	0.000000	0.231518
75%	46.500000	1.000000e+00	1.000000e+00	90.000000	0.685590	1.000000	0.521286	0.363210
max	60.900000	1.000000e+00	1.918334e+00	229.381725	1.000000	1.000000	1.000000	1.000000

In [41]:
```
pb3_2.describe()
```

Out[41]:

	Age	SibSp	Parch	Fare	Age_2	SibSp_2	Parch_2	Fare_2
count	183.000000	1.830000e+02	1.830000e+02	183.000000	183.000000	183.000000	183.000000	183.000000
mean	35.889690	3.955373e-01	4.412416e-01	74.762974	0.453923	0.395537	0.230013	0.293597
std	12.929136	4.861124e-01	6.666807e-01	58.137731	0.282296	0.486112	0.347531	0.265613
min	15.100000	2.220446e-16	-3.330669e-16	10.500000	0.000000	0.000000	0.000000	0.000000
25%	25.000000	2.220446e-16	-3.330669e-16	30.000000	0.216157	0.000000	0.000000	0.089089
50%	35.692532	2.220446e-16	-3.330669e-16	61.175000	0.449619	0.000000	0.000000	0.231518
75%	46.500000	1.000000e+00	1.000000e+00	90.000000	0.685590	1.000000	0.521286	0.363210
max	60.900000	1.000000e+00	1.918334e+00	229.381725	1.000000	1.000000	1.000000	1.000000

In [48]:
```
pc_2.describe()
```

Out[48]:

	Age	SibSp	Parch	Fare	Age_2	SibSp_2	Parch_2	Fare_2
count	183.000000	1.830000e+02	1.830000e+02	183.000000	183.000000	183.000000	183.000000	183.000000
mean	35.889690	3.955373e-01	4.412416e-01	74.762974	0.453923	0.395537	0.230013	0.293597
std	12.929136	4.861124e-01	6.666807e-01	58.137731	0.282296	0.486112	0.347531	0.265613
min	15.100000	2.220446e-16	-3.330669e-16	10.500000	0.000000	0.000000	0.000000	0.000000
25%	25.000000	2.220446e-16	-3.330669e-16	30.000000	0.216157	0.000000	0.000000	0.089089
50%	35.692532	2.220446e-16	-3.330669e-16	61.175000	0.449619	0.000000	0.000000	0.231518
75%	46.500000	1.000000e+00	1.000000e+00	90.000000	0.685590	1.000000	0.521286	0.363210
max	60.900000	1.000000e+00	1.918334e+00	229.381725	1.000000	1.000000	1.000000	1.000000

In [43]:
```
pd_2.describe()
```

Out[43]:

	Age	SibSp	Parch	Fare	Age_2	SibSp_2	Parch_2	Fare_2
count	183.000000	1.830000e+02	1.830000e+02	183.000000	183.000000	183.000000	183.000000	183.000000
mean	35.889690	3.955373e-01	4.412416e-01	74.762974	0.453923	0.395537	0.230013	0.293597
std	12.929136	4.861124e-01	6.666807e-01	58.137731	0.282296	0.486112	0.347531	0.265613
min	15.100000	2.220446e-16	-3.330669e-16	10.500000	0.000000	0.000000	0.000000	0.000000
25%	25.000000	2.220446e-16	-3.330669e-16	30.000000	0.216157	0.000000	0.000000	0.089089
50%	35.692532	2.220446e-16	-3.330669e-16	61.175000	0.449619	0.000000	0.000000	0.231518
75%	46.500000	1.000000e+00	1.000000e+00	90.000000	0.685590	1.000000	0.521286	0.363210
max	60.900000	1.000000e+00	1.918334e+00	229.381725	1.000000	1.000000	1.000000	1.000000

In [44]:
```
pe_2.describe()
```

Out[44]:

	Age	SibSp	Parch	Fare	Age_2	SibSp_2	Parch_2	Fare_2
count	183.000000	1.830000e+02	1.830000e+02	183.000000	183.000000	183.000000	183.000000	183.000000
mean	35.889690	3.955373e-01	4.412416e-01	74.762974	0.453923	0.395537	0.230013	0.293597
std	12.929136	4.861124e-01	6.666807e-01	58.137731	0.282296	0.486112	0.347531	0.265613
min	15.100000	2.220446e-16	-3.330669e-16	10.500000	0.000000	0.000000	0.000000	0.000000
25%	25.000000	2.220446e-16	-3.330669e-16	30.000000	0.216157	0.000000	0.000000	0.089089
50%	35.692532	2.220446e-16	-3.330669e-16	61.175000	0.449619	0.000000	0.000000	0.231518
75%	46.500000	1.000000e+00	1.000000e+00	90.000000	0.685590	1.000000	0.521286	0.363210
max	60.900000	1.000000e+00	1.918334e+00	229.381725	1.000000	1.000000	1.000000	1.000000

In [45]:
```
pf_2.describe()
```

Out[45]:

	Age	SibSp	Parch	Fare	Age_2	SibSp_2	Parch_2	Fare_2
count	183.000000	1.830000e+02	1.830000e+02	183.000000	183.000000	183.000000	183.000000	183.000000
mean	35.889690	3.955373e-01	4.412416e-01	74.762974	0.453923	0.395537	0.230013	0.293597
std	12.929136	4.861124e-01	6.666807e-01	58.137731	0.282296	0.486112	0.347531	0.265613
min	15.100000	2.220446e-16	-3.330669e-16	10.500000	0.000000	0.000000	0.000000	0.000000
25%	25.000000	2.220446e-16	-3.330669e-16	30.000000	0.216157	0.000000	0.000000	0.089089
50%	35.692532	2.220446e-16	-3.330669e-16	61.175000	0.449619	0.000000	0.000000	0.231518
75%	46.500000	1.000000e+00	1.000000e+00	90.000000	0.685590	1.000000	0.521286	0.363210
max	60.900000	1.000000e+00	1.918334e+00	229.381725	1.000000	1.000000	1.000000	1.000000

In [49]:
```
df_3.describe()
```

Out[49]:

	PassengerId	Survived	Pclass	Age	SibSp	Parch	Fare	Age_3	
count	183.000000	183.000000	183.000000	183.000000	183.000000	183.000000	183.000000	183.000000	183
mean	455.366120	0.672131	1.191257	35.674426	0.464481	0.475410	78.682469	0.445930	0
std	247.052476	0.470725	0.515187	15.643866	0.644159	0.754617	76.347843	0.195548	0
min	2.000000	0.000000	1.000000	0.920000	0.000000	0.000000	0.000000	0.011500	0
25%	263.500000	0.000000	1.000000	24.000000	0.000000	0.000000	29.700000	0.300000	0
50%	457.000000	1.000000	1.000000	36.000000	0.000000	0.000000	57.000000	0.450000	0
75%	676.000000	1.000000	1.000000	47.500000	1.000000	1.000000	90.000000	0.593750	0
max	890.000000	1.000000	3.000000	80.000000	3.000000	4.000000	512.329200	1.000000	1

In [50]:
```
pb1_3.describe()
```

Out[50]:

	Age	SibSp	Parch	Fare	Age_3	SibSp_3	Parch_3	Fare
count	183.000000	1.830000e+02	1.830000e+02	183.000000	183.000000	1.830000e+02	1.830000e+02	183.000
mean	35.889690	3.955373e-01	4.412416e-01	74.762974	0.589322	3.955373e-01	2.300129e-01	0.325
std	12.929136	4.861124e-01	6.666807e-01	58.137731	0.212301	4.861124e-01	3.475311e-01	0.253
min	15.100000	2.220446e-16	-3.330669e-16	10.500000	0.247947	2.220446e-16	-1.736230e-16	0.045
25%	25.000000	2.220446e-16	-3.330669e-16	30.000000	0.410509	2.220446e-16	-1.736230e-16	0.130
50%	35.692532	2.220446e-16	-3.330669e-16	61.175000	0.586084	2.220446e-16	-1.736230e-16	0.266
75%	46.500000	1.000000e+00	1.000000e+00	90.000000	0.763547	1.000000e+00	5.212856e-01	0.392
max	60.900000	1.000000e+00	1.918334e+00	229.381725	1.000000	1.000000e+00	1.000000e+00	1.000

In [51]:
```
pb2_3.describe()
```

Out[51]:

	Age	SibSp	Parch	Fare	Age_3	SibSp_3	Parch_3	Fare
count	183.000000	1.830000e+02	1.830000e+02	183.000000	183.000000	1.830000e+02	1.830000e+02	183.000
mean	35.889690	3.955373e-01	4.412416e-01	74.762974	0.589322	3.955373e-01	2.300129e-01	0.325
std	12.929136	4.861124e-01	6.666807e-01	58.137731	0.212301	4.861124e-01	3.475311e-01	0.253
min	15.100000	2.220446e-16	-3.330669e-16	10.500000	0.247947	2.220446e-16	-1.736230e-16	0.045
25%	25.000000	2.220446e-16	-3.330669e-16	30.000000	0.410509	2.220446e-16	-1.736230e-16	0.130
50%	35.692532	2.220446e-16	-3.330669e-16	61.175000	0.586084	2.220446e-16	-1.736230e-16	0.266
75%	46.500000	1.000000e+00	1.000000e+00	90.000000	0.763547	1.000000e+00	5.212856e-01	0.392
max	60.900000	1.000000e+00	1.918334e+00	229.381725	1.000000	1.000000e+00	1.000000e+00	1.000

In [52]:
```
pb3_3.describe()
```

Out[52]:

	Age	SibSp	Parch	Fare	Age_3	SibSp_3	Parch_3	Fare
count	183.000000	1.830000e+02	1.830000e+02	183.000000	183.000000	1.830000e+02	1.830000e+02	183.0000
mean	35.889690	3.955373e-01	4.412416e-01	74.762974	0.589322	3.955373e-01	2.300129e-01	0.3259
std	12.929136	4.861124e-01	6.666807e-01	58.137731	0.212301	4.861124e-01	3.475311e-01	0.2534
min	15.100000	2.220446e-16	-3.330669e-16	10.500000	0.247947	2.220446e-16	-1.736230e-16	0.0457
25%	25.000000	2.220446e-16	-3.330669e-16	30.000000	0.410509	2.220446e-16	-1.736230e-16	0.1307
50%	35.692532	2.220446e-16	-3.330669e-16	61.175000	0.586084	2.220446e-16	-1.736230e-16	0.2666
75%	46.500000	1.000000e+00	1.000000e+00	90.000000	0.763547	1.000000e+00	5.212856e-01	0.3923
max	60.900000	1.000000e+00	1.918334e+00	229.381725	1.000000	1.000000e+00	1.000000e+00	1.0000

In [53]:
```
pc_3.describe()
```

Out[53]:

	Age	SibSp	Parch	Fare	Age_3	SibSp_3	Parch_3	Fare
count	183.000000	1.830000e+02	1.830000e+02	183.000000	183.000000	1.830000e+02	1.830000e+02	183.0000
mean	35.889690	3.955373e-01	4.412416e-01	74.762974	0.589322	3.955373e-01	2.300129e-01	0.3259
std	12.929136	4.861124e-01	6.666807e-01	58.137731	0.212301	4.861124e-01	3.475311e-01	0.2534
min	15.100000	2.220446e-16	-3.330669e-16	10.500000	0.247947	2.220446e-16	-1.736230e-16	0.0457
25%	25.000000	2.220446e-16	-3.330669e-16	30.000000	0.410509	2.220446e-16	-1.736230e-16	0.1307
50%	35.692532	2.220446e-16	-3.330669e-16	61.175000	0.586084	2.220446e-16	-1.736230e-16	0.2666
75%	46.500000	1.000000e+00	1.000000e+00	90.000000	0.763547	1.000000e+00	5.212856e-01	0.3923
max	60.900000	1.000000e+00	1.918334e+00	229.381725	1.000000	1.000000e+00	1.000000e+00	1.0000

In [54]:
```
pd_3.describe()
```

Out[54]:

	Age	SibSp	Parch	Fare	Age_3	SibSp_3	Parch_3	Fare
count	183.000000	1.830000e+02	1.830000e+02	183.000000	183.000000	1.830000e+02	1.830000e+02	183.0000
mean	35.889690	3.955373e-01	4.412416e-01	74.762974	0.589322	3.955373e-01	2.300129e-01	0.3259
std	12.929136	4.861124e-01	6.666807e-01	58.137731	0.212301	4.861124e-01	3.475311e-01	0.2534
min	15.100000	2.220446e-16	-3.330669e-16	10.500000	0.247947	2.220446e-16	-1.736230e-16	0.0457
25%	25.000000	2.220446e-16	-3.330669e-16	30.000000	0.410509	2.220446e-16	-1.736230e-16	0.1307
50%	35.692532	2.220446e-16	-3.330669e-16	61.175000	0.586084	2.220446e-16	-1.736230e-16	0.2666
75%	46.500000	1.000000e+00	1.000000e+00	90.000000	0.763547	1.000000e+00	5.212856e-01	0.3923
max	60.900000	1.000000e+00	1.918334e+00	229.381725	1.000000	1.000000e+00	1.000000e+00	1.0000

In [55]:
```
pe_3.describe()
```

Out[55]:

	Age	SibSp	Parch	Fare	Age_3	SibSp_3	Parch_3	Fare
count	183.000000	1.830000e+02	1.830000e+02	183.000000	183.000000	1.830000e+02	1.830000e+02	183.000(
mean	35.889690	3.955373e-01	4.412416e-01	74.762974	0.589322	3.955373e-01	2.300129e-01	0.3259
std	12.929136	4.861124e-01	6.666807e-01	58.137731	0.212301	4.861124e-01	3.475311e-01	0.2534
min	15.100000	2.220446e-16	-3.330669e-16	10.500000	0.247947	2.220446e-16	-1.736230e-16	0.0457
25%	25.000000	2.220446e-16	-3.330669e-16	30.000000	0.410509	2.220446e-16	-1.736230e-16	0.1307
50%	35.692532	2.220446e-16	-3.330669e-16	61.175000	0.586084	2.220446e-16	-1.736230e-16	0.2666
75%	46.500000	1.000000e+00	1.000000e+00	90.000000	0.763547	1.000000e+00	5.212856e-01	0.3923
max	60.900000	1.000000e+00	1.918334e+00	229.381725	1.000000	1.000000e+00	1.000000e+00	1.0000

In [56]:
```
pf_3.describe()
```

Out[56]:

	Age	SibSp	Parch	Fare	Age_3	SibSp_3	Parch_3	Fare
count	183.000000	1.830000e+02	1.830000e+02	183.000000	183.000000	1.830000e+02	1.830000e+02	183.000(
mean	35.889690	3.955373e-01	4.412416e-01	74.762974	0.589322	3.955373e-01	2.300129e-01	0.3259
std	12.929136	4.861124e-01	6.666807e-01	58.137731	0.212301	4.861124e-01	3.475311e-01	0.2534
min	15.100000	2.220446e-16	-3.330669e-16	10.500000	0.247947	2.220446e-16	-1.736230e-16	0.0457
25%	25.000000	2.220446e-16	-3.330669e-16	30.000000	0.410509	2.220446e-16	-1.736230e-16	0.1307
50%	35.692532	2.220446e-16	-3.330669e-16	61.175000	0.586084	2.220446e-16	-1.736230e-16	0.2666
75%	46.500000	1.000000e+00	1.000000e+00	90.000000	0.763547	1.000000e+00	5.212856e-01	0.3923
max	60.900000	1.000000e+00	1.918334e+00	229.381725	1.000000	1.000000e+00	1.000000e+00	1.0000

*Los datos de la normalizacion por standarization se definian por un rango de 0-1 y una desviacion de 1, mientras los otros dos modelos se normalizan por el minimo y maximo manejado en cada dato.La normalizacion es mas beneficiosa cuando se conoce la distribucion de los datos (o se puede determinar) mientras la standarization no esta tan atada al rango de datos por lo que no se acaba con los outliers