TITANIC - MACHINE LEARNING FROM DISASTER

DATASET UNDERSTANDING

Dataset yang digunakan kali ini berasal dari keggle (https://www.kaggle.com/c/titanic/overview)

Data yang ada merupakan data penumpang kapal titanic dengan beberapa fitur.

	PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare	Cabin	Embarked
432	432	1	3	Thomeycroft, Mrs. Percival (Florence Kate White)	female		1	0	376564	16.1		S
433	433	1	2	Louch, Mrs. Charles Alexander (Alice Adelaide Slow)	female	42	1	0	SC/AH 3085	26		S
434	434	0	3	Kallio, Mr. Nikolai Erland	male	17	0	0	STON/O 2. 3101274	7.125		S
435	435	0	1	Silvey, Mr. William Baird	male	50	1	0	13507	55.9	E44	S
436	436	1	1	Carter, Miss. Lucile Polk	female	14	1	2	113760	120	B96 B98	S
437	437	0	3	Ford, Miss. Doolina Margaret "Daisy"	female	21	2	2	W./C. 6608	34.375		S
438	438	1	2	Richards, Mrs. Sidney (Emily Hocking)	female	24	2	3	29106	18.75		S
439	439	0	1	Fortune, Mr. Mark	male	64	1	4	19950	263	C23 C25 C27	S
440	440	0	2	Kvillner, Mr. Johan Henrik Johannesson	male	31	0	0	C.A. 18723	10.5		S
441	441	1	2	Hart, Mrs. Benjamin (Esther Ada Bloomfield)	female	45	1	1	F.C.C. 13529	26.25		S

Pada challenge kali ini di Kaggle, diminta untuk memprediksi status survived dari file test.csv yang disediakan.

FITUR YANG ADA PADA DATASET

Variable	Definition	Key
survival	Survival	0 = No, 1 = Yes
pclass	Ticket class	1 = 1st, 2 = 2nd, 3 = 3rd
sex	Sex	
Age	Age in years	
sibsp	# of siblings / spouses aboard the Titanic	
parch	# of parents / children aboard the Titanic	
ticket	Ticket number	
fare	Passenger fare	
cabin	Cabin number	
embarked	Port of Embarkation	C = Cherbourg, Q = Queenstown, S = Southampton

INFORMASI MENGENAI DATASET

Untuk train.csv berisi 891 baris dan 12 kolom sedangkan untuk test.csv terdapat 418 baris dan 11 kolom

```
[4]  print("Dimensi train set", train_df.shape)
     print("Dimensi tes set", test_df.shape)
```

```
Dimensi train set (891, 12)
Dimensi tes set (418, 11)
```

```
# informasi train dataset
train_df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 891 entries, 0 to 890
Data columns (total 12 columns):
 #   Column       Non-Null Count  Dtype
---  ------       --------------  -----
 0   PassengerId  891 non-null    int64
 1   Survived     891 non-null    int64
 2   Pclass       891 non-null    int64
 3   Name         891 non-null    object
 4   Sex          891 non-null    object
 5   Age          714 non-null    float64
 6   SibSp        891 non-null    int64
 7   Parch        891 non-null    int64
 8   Ticket       891 non-null    object
 9   Fare         891 non-null    float64
 10  Cabin        204 non-null    object
 11  Embarked     889 non-null    object
dtypes: float64(2), int64(5), object(5)
memory usage: 83.7+ KB
```

```
# informasi test dataset
test_df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 418 entries, 0 to 417
Data columns (total 11 columns):
 #   Column       Non-Null Count  Dtype
---  ------       --------------  -----
 0   PassengerId  418 non-null    int64
 1   Pclass       418 non-null    int64
 2   Name         418 non-null    object
 3   Sex          418 non-null    object
 4   Age          332 non-null    float64
 5   SibSp        418 non-null    int64
 6   Parch        418 non-null    int64
 7   Ticket       418 non-null    object
 8   Fare         417 non-null    float64
 9   Cabin        91 non-null     object
 10  Embarked     418 non-null    object
dtypes: float64(2), int64(4), object(5)
memory usage: 36.0+ KB
```

MISSING VALUES

```
# train dataset
train_df.isnull().sum().sort_values(ascending=False)
```

```
Cabin          687
Age            177
Embarked         2
PassengerId      0
Survived         0
Pclass           0
Name             0
Sex              0
SibSp            0
Parch            0
Ticket           0
Fare             0
dtype: int64
```

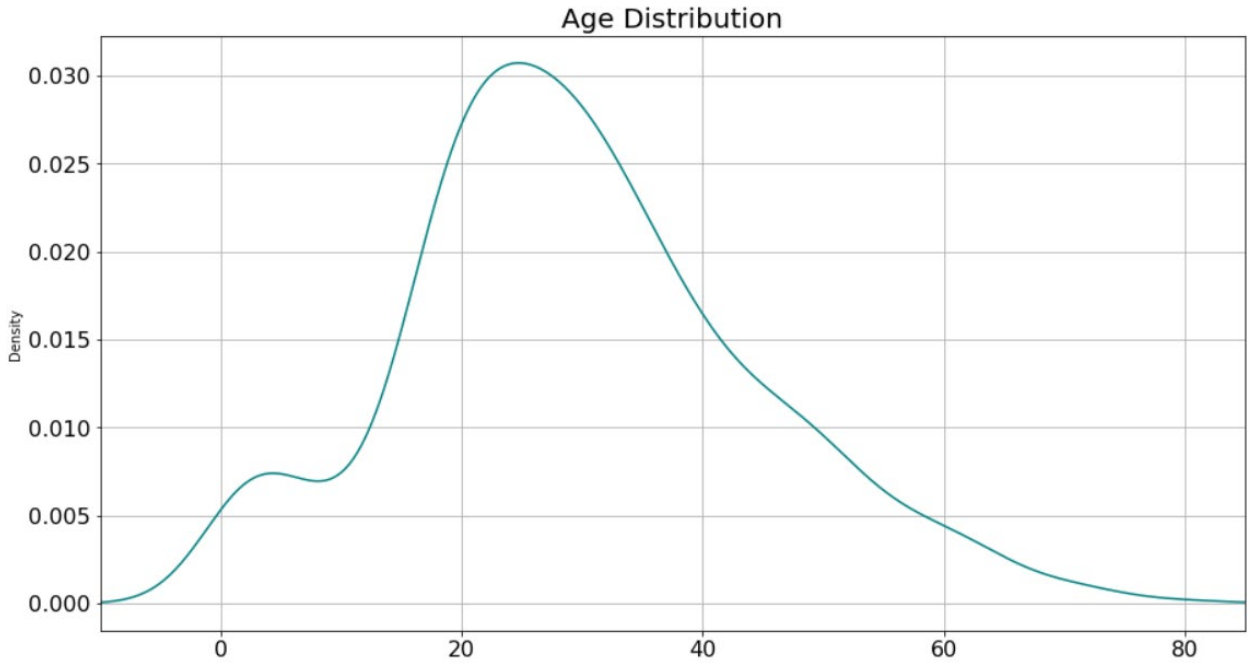
```
# test dataset
test_df.isnull().sum().sort_values(ascending=False)
```

```
Cabin          327
Age             86
Fare             1
PassengerId      0
Pclass           0
Name             0
Sex              0
SibSp            0
Parch            0
Ticket           0
Embarked         0
dtype: int64
```

PENANGANAN MISSING VALUES UNTUK TRAIN DATASET

Kolom "Age"

```
[9]  plt.figure(figsize=(15,8))
     train_df["Age"].plot(kind='density', color='teal', fontsize=16)
     plt.xlim(-10,85)
     plt.grid()
     plt.title("Age Distribution", fontsize=20)
     plt.show()
```

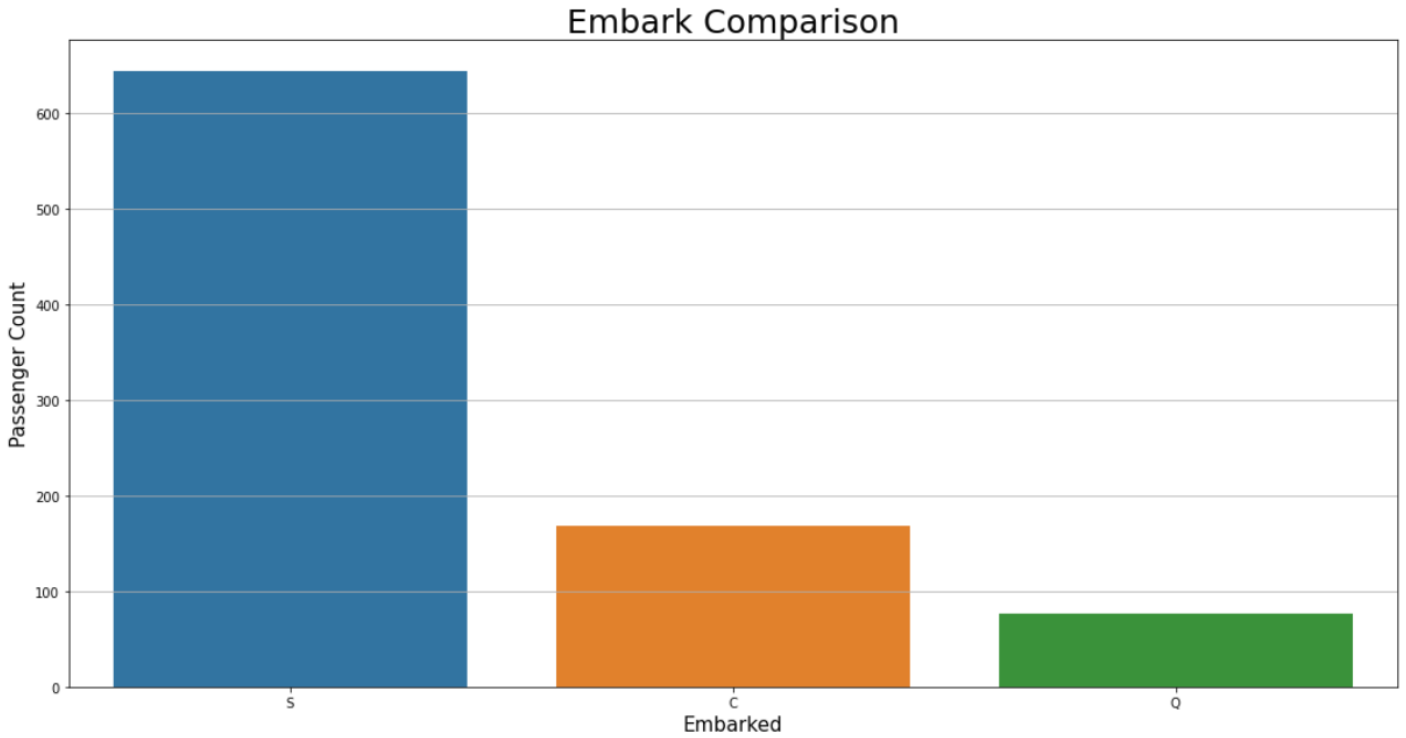


Karena distribusi dari "Age" cenderung skewed dan bersifat numerik maka missing values dalam kolom "Age" akan diisi oleh median dari data "Age"

```
train_df["Age"].fillna(train_df["Age"].median(skipna=True), inplace=True)
```

Kolom "Embarked"

```
[11] plt.figure(figsize=(15,8))
     sns.countplot(data=train_df, x='Embarked')
     plt.grid(axis='y')
     plt.title("Embark Comparison", fontsize=25)
     plt.xlabel("Embarked",fontsize=15)
     plt.ylabel("Passenger Count",fontsize=15)
     plt.tight_layout()
```



Kolom "Embarked" bersifat kategorikal sehingga missing values di dalamnya akan diisi oleh nilai modus dari data tersebut

```
train_df['Embarked'].fillna(train_df["Embarked"].mode()[0], inplace=True)
```

Kolom "Cabin"

```
[13] percent_cabin_null = train_df["Cabin"].isnull().sum() / train_df["Cabin"].size *100
     print("Persentase missing values dalam kolom kabin berjumlah", percent_cabin_null, "%")
```

```
Persentase missing values dalam kolom kabin berjumlah 77.10437710437711 %
```

Karena missing values dalam kolom "Cabin" terlalu banyak maka kolom kabin akan dihilangkan

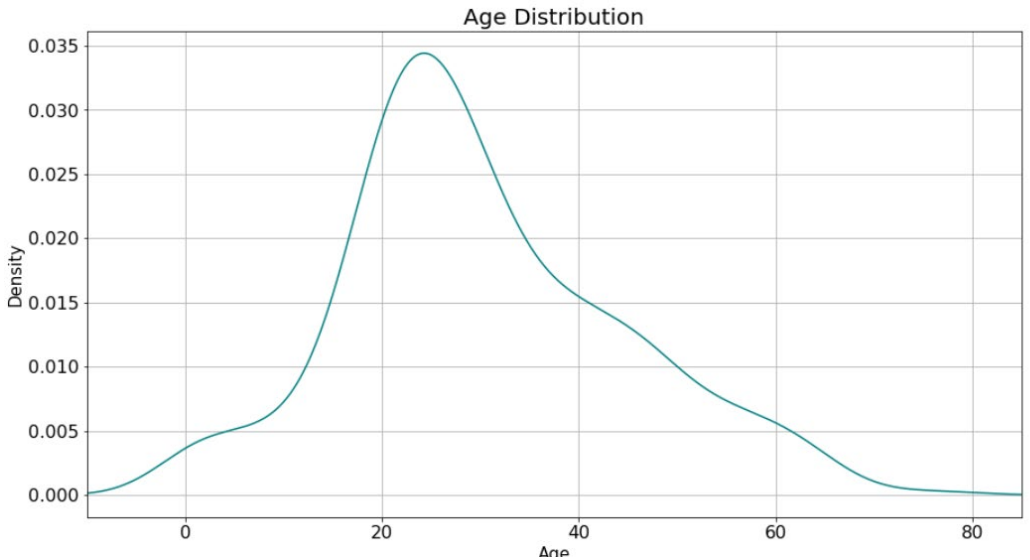
```
train_df.drop('Cabin', axis=1, inplace=True)
```

PENANGANAN MISSING VALUES UNTUK TEST DATASET

Kolom "Age"

```
[15] plt.figure(figsize=(15,8))
     test_df["Age"].plot(kind='density', color='teal',fontsize=16)
     plt.xlim(-10,85)
     plt.xlabel("Age",fontsize=15)
     plt.ylabel("Density",fontsize=15)
     plt.grid()
     plt.title("Age Distribution",fontsize=20)

     plt.show()
```



Sama seperti pada train dataset, missing values kolom "Age" pada test dataset pun diisi dengan nilai median

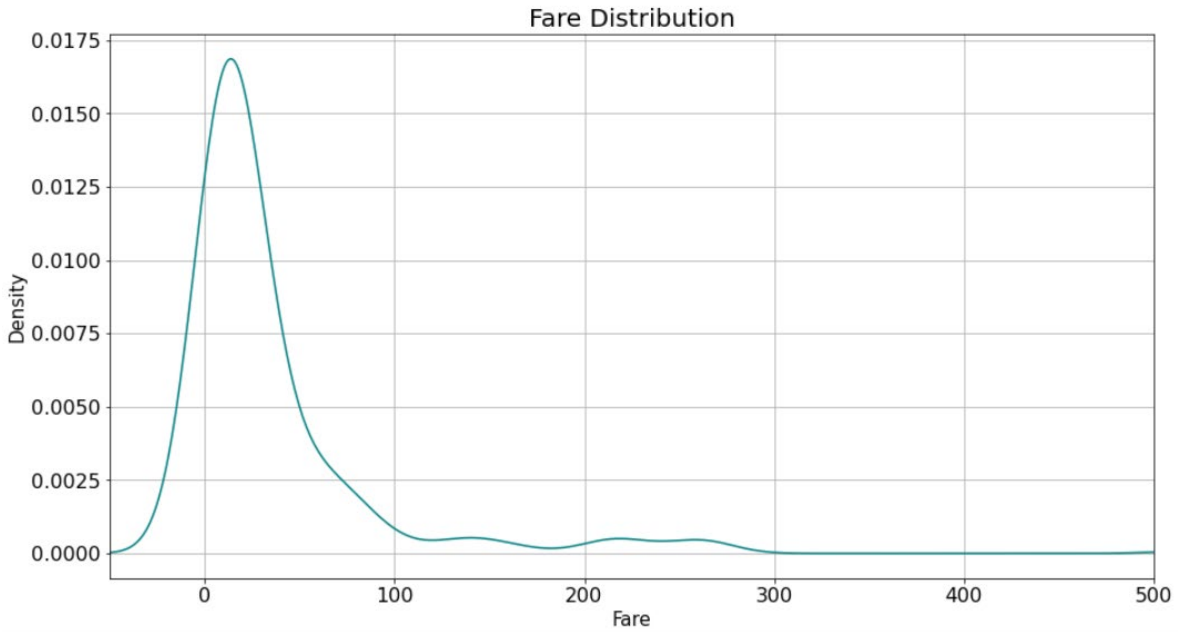
```
test_df["Age"].fillna(test_df["Age"].median(skipna=True), inplace=True)
```

Kolom "Fare"

```
[17] test_df["Fare"].max()
```

```
512.3292
```

```
[18] plt.figure(figsize=(15,8))
     test_df["Fare"].plot(kind='density', color='teal',fontsize=16)
     plt.xlabel("Fare",fontsize=15)
     plt.ylabel('Density',fontsize=15)
     plt.xlim(-50,500)
     plt.grid()
     plt.title("Fare Distribution", fontsize=20)
     plt.show()
```



Fill Missing Value

```
[19] test_df['Fare'].fillna(test_df["Fare"].median(), inplace=True)
```

Kolom "Cabin"

```
[20] percent_cabin_null = test_df["Cabin"].isnull().sum() / test_df["Cabin"].size *100
     print("Persentase missing values dalam kolom kabin berjumlah", percent_cabin_null, "%")
```

```
Persentase missing values dalam kolom kabin berjumlah 78.22966507177034 %
```

sama seperti train dataset nilai missing values "Cabin" pada test dataset terlalu banyak sehingga dihilangkan

```
test_df.drop('Cabin', axis=1, inplace=True)
```

HASIL PENANGANAN MISSING VALUES

```
# Train Dataset
train_df.isnull().sum().sort_values(ascending=False)
```

```
PassengerId    0
Survived       0
Pclass         0
Name           0
Sex            0
Age            0
SibSp          0
Parch          0
Ticket         0
Fare           0
Embarked       0
dtype: int64
```

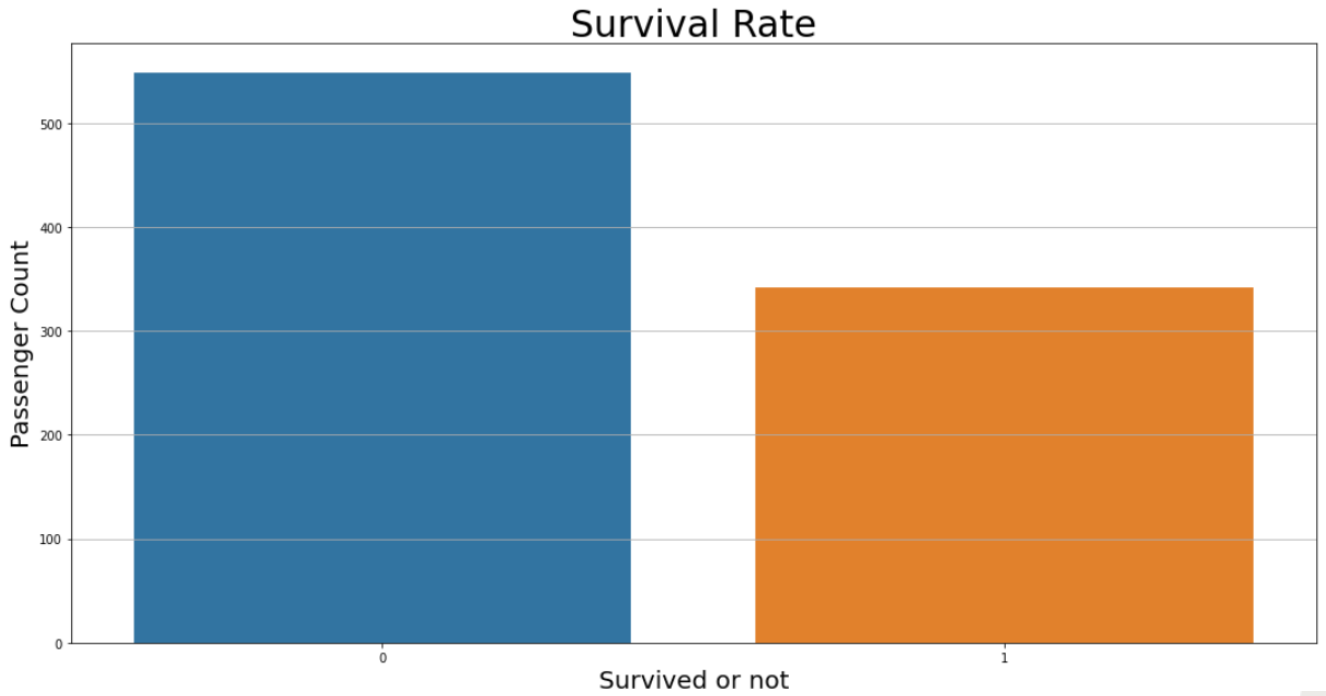
```
# Test Dataset
test_df.isnull().sum().sort_values(ascending=False)
```

```
PassengerId    0
Pclass         0
Name           0
Sex            0
Age            0
SibSp          0
Parch          0
Ticket         0
Fare           0
Embarked       0
dtype: int64
```

DATA VISUALIZATION

Perbandingan jumlah penumpang selamat

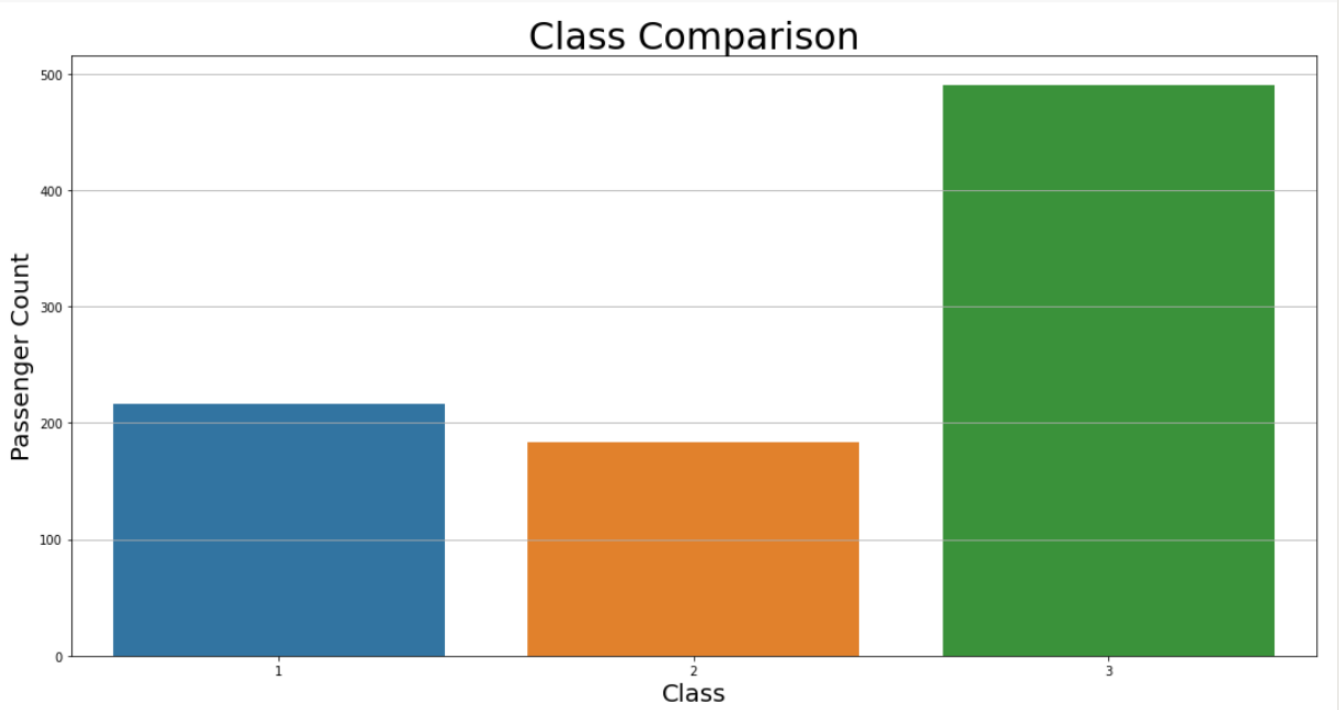
```
[24] plt.figure(figsize=(15,8))
     sns.countplot(data=train_df, x='Survived')
     plt.grid(axis='y')
     plt.xlabel('Survived or not',fontsize=20)
     plt.ylabel('Passenger Count',fontsize=20)
     plt.title("Survival Rate",fontsize=30)
     plt.tight_layout()
```



Dari sini terlihat bahwa jumlah penumpang yang tidak selamat lebih banyak daripada jumlah penumpang yang selamat

- Perbandingan penumpang berdasarkan kelas

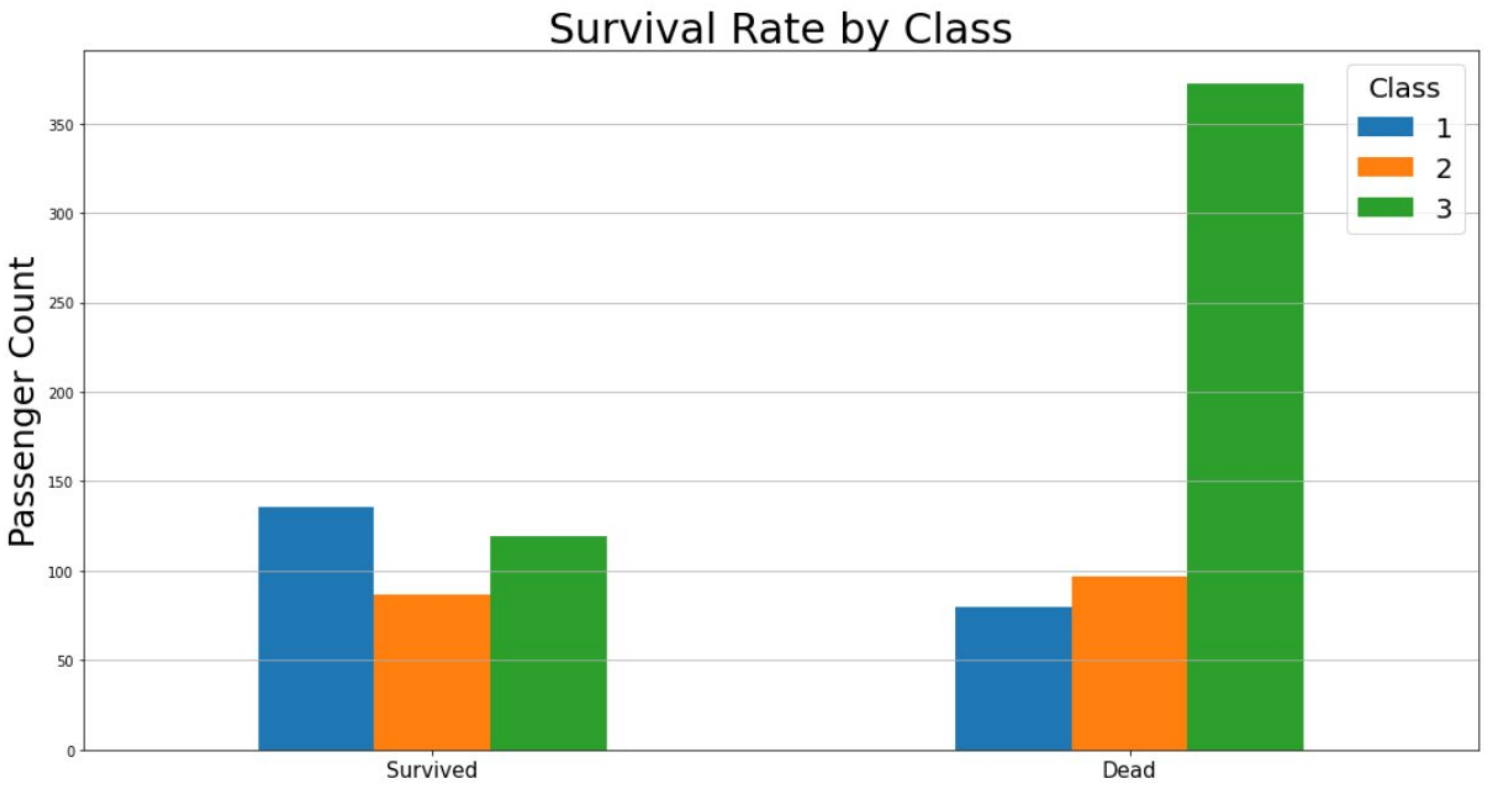
```
[25] plt.figure(figsize=(15,8))
     sns.countplot(data=train_df, x='Pclass')
     plt.grid(axis='y')
     plt.xlabel('Class',fontsize=20)
     plt.ylabel('Passenger Count',fontsize=20)
     plt.title("Class Comparison",fontsize=30)
     plt.tight_layout()
```



Jumlah penumpang kelas 3 lebih banyak dari kedua kelas lainnya

```python
def bar_chart(feature):
  survived = train_df[train_df["Survived"]==1][feature].value_counts()
  dead = train_df[train_df["Survived"]==0][feature].value_counts()
  df=pd.DataFrame([survived,dead])
  df.index=["Survived","Dead"]
  df.plot(kind="bar", stacked = False, figsize=(15,8))
  plt.xticks(rotation=0,fontsize=15)
  plt.grid(axis='y')
  plt.ylabel('Passenger Count',fontsize=25)
```

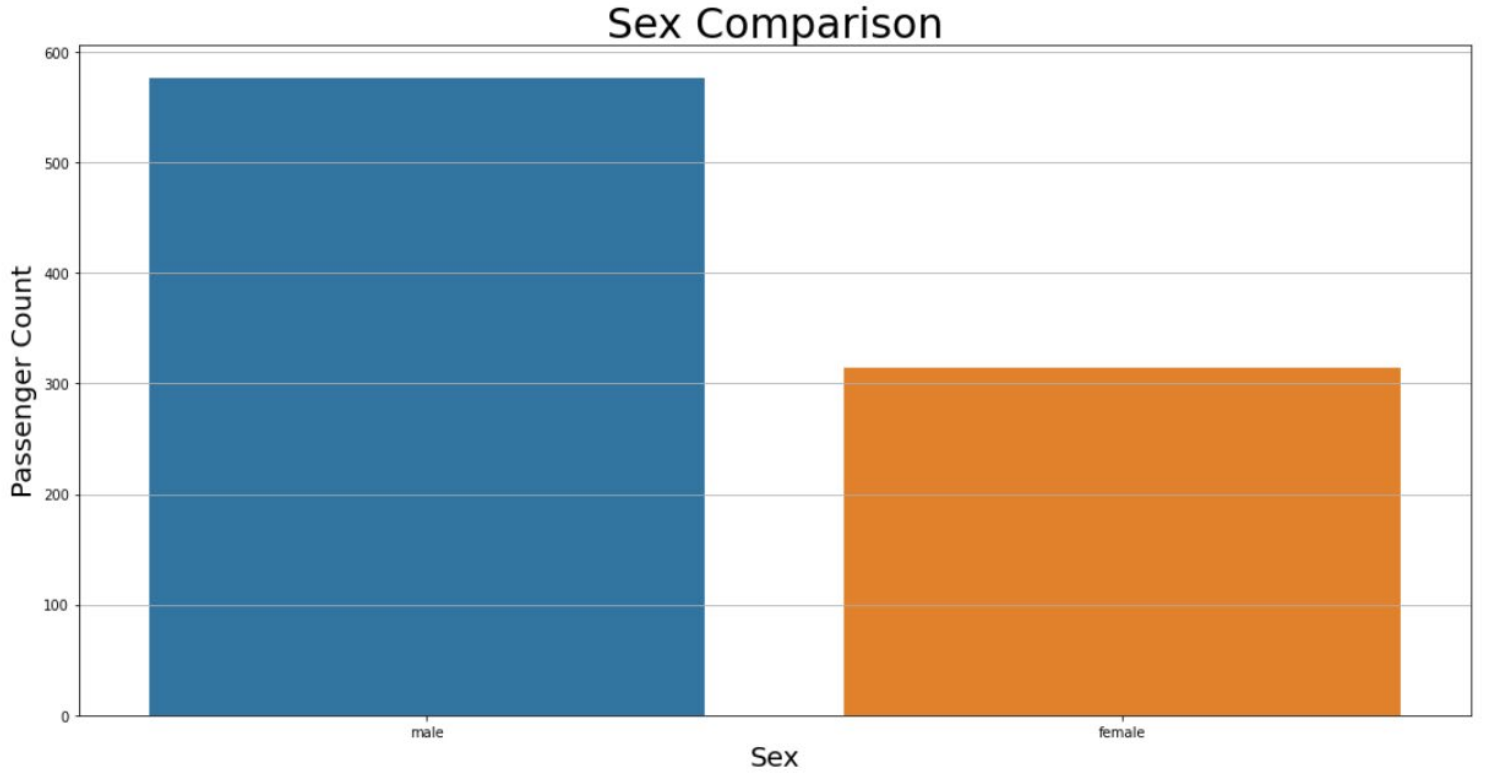
```python
bar_chart("Pclass")
plt.title("Survival Rate by Class",fontsize=30)
plt.legend(title="Class",title_fontsize=20,fontsize=20)
plt.tight_layout()
```



Jumlah kematian penumpang berdasarkan kelas paling banyak teradapat pada kelas 3

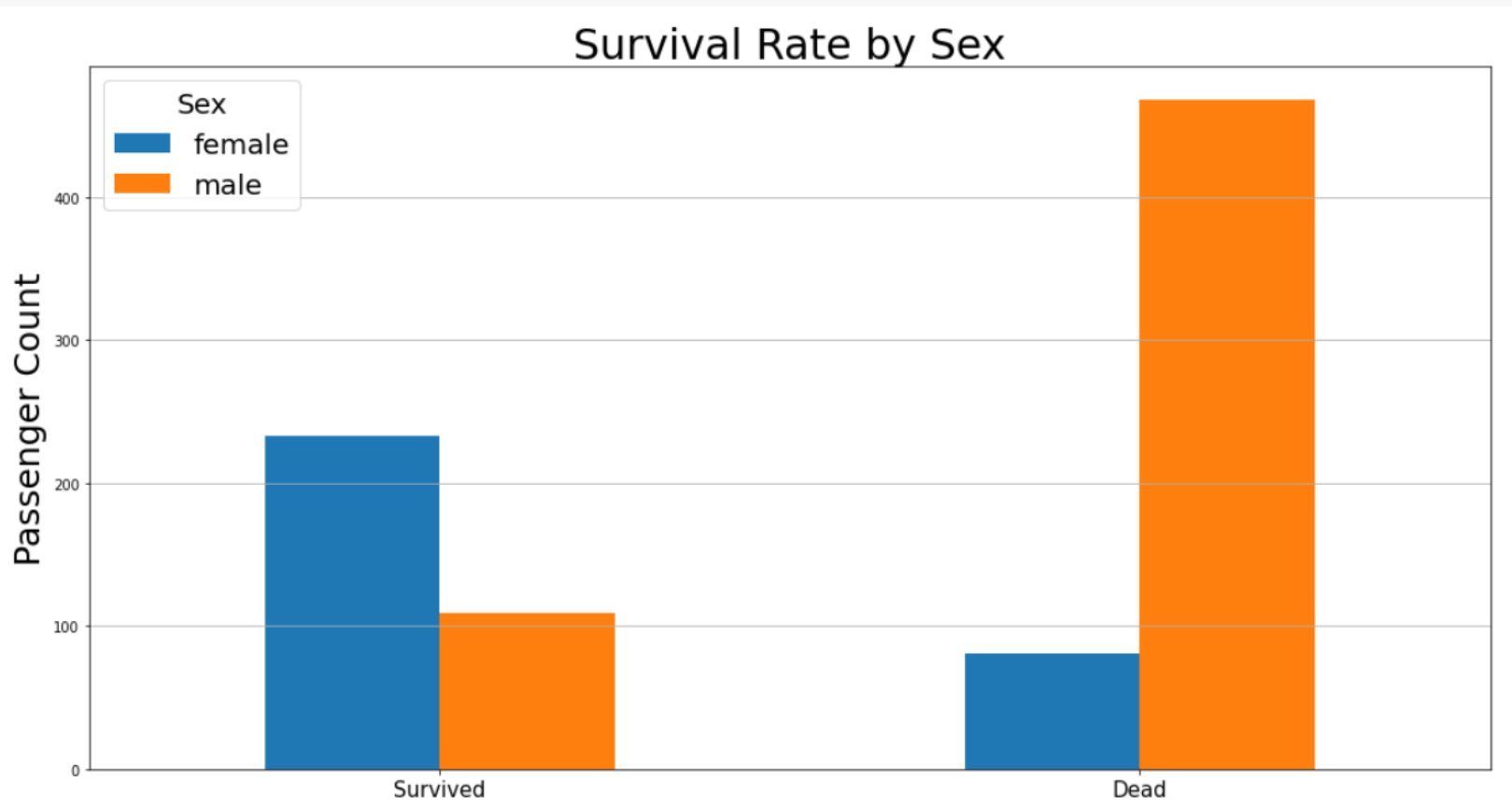
Perbandingan jenis kelamin penumpang

```
[28] plt.figure(figsize=(15,8))
     sns.countplot(data=train_df, x='Sex')
     plt.grid(axis='y')
     plt.xlabel('Sex',fontsize=20)
     plt.ylabel('Passenger Count',fontsize=20)
     plt.title("Sex Comparison",fontsize=30)
     plt.tight_layout()
```



Jumlah penumpang pria lebih banyak dari penumpang wanita

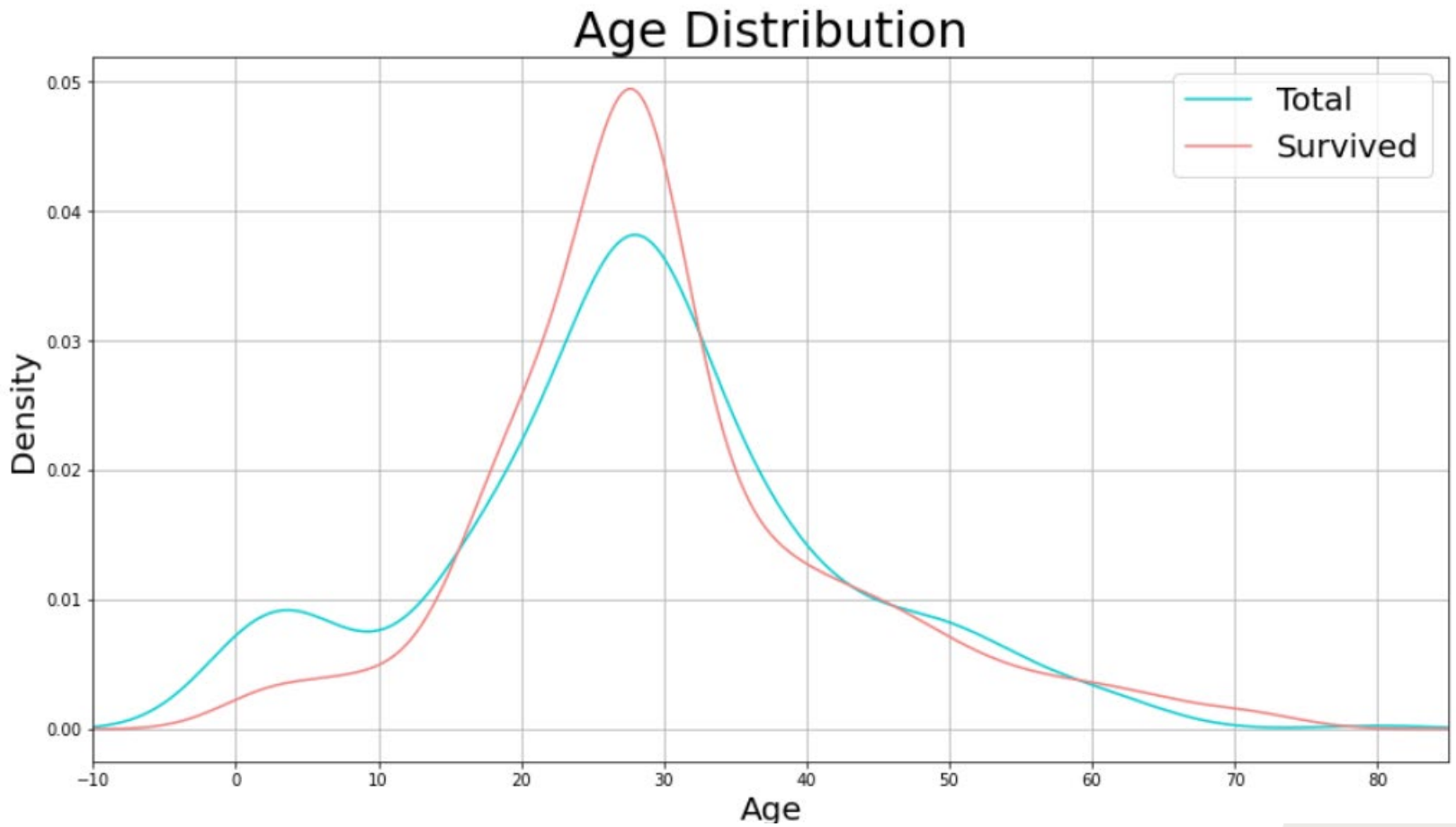
```
bar_chart("Sex")
plt.title("Survival Rate by Sex",fontsize=30)
plt.legend(title="Sex",title_fontsize=20,fontsize=20)
plt.tight_layout()
```



Jumlah penumpang pria yang tewas lebih signifikan daripada penumpang wanita

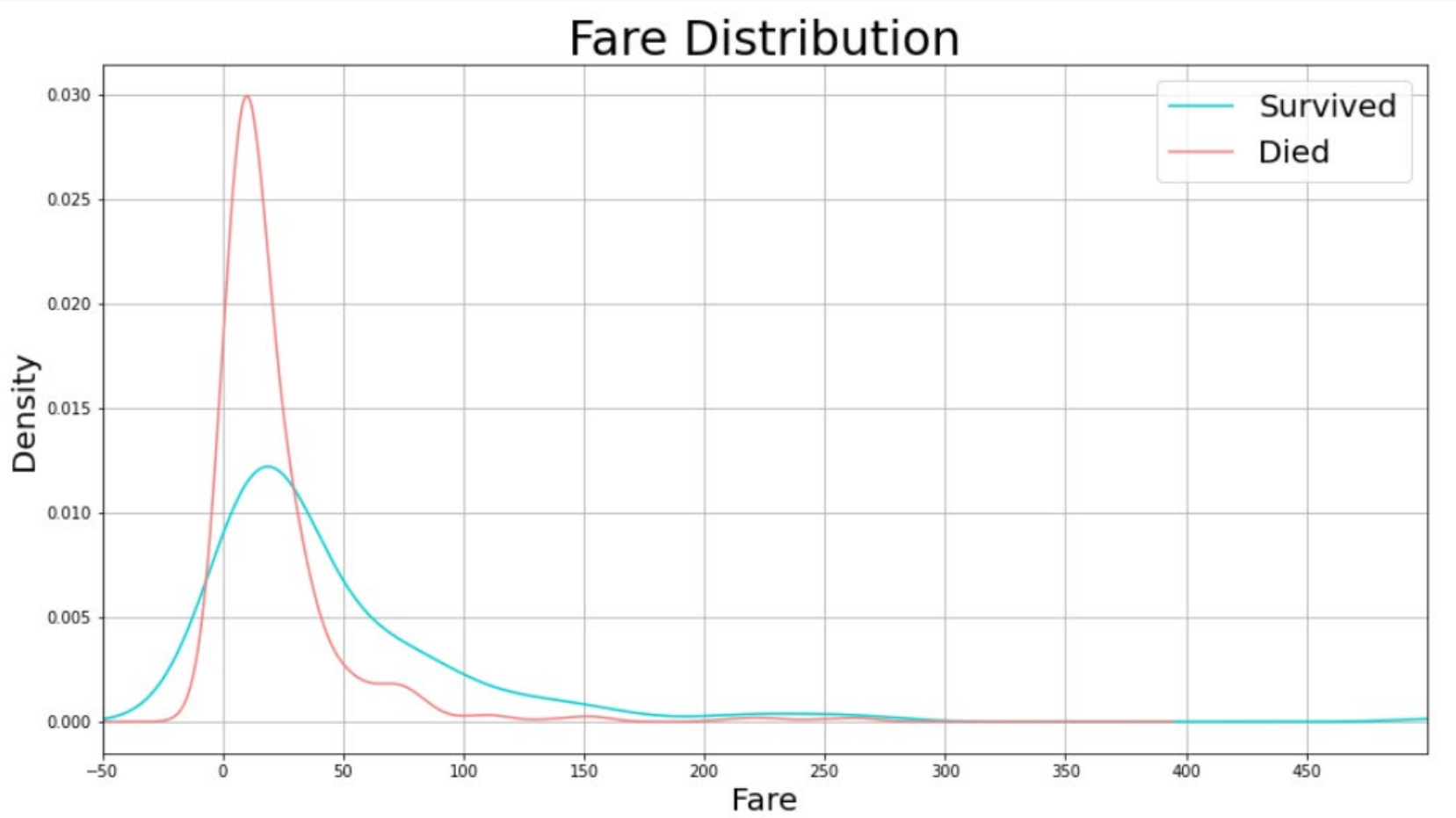
Distribusi umur

```
[30] plt.figure(figsize=(15,8))
     train_df["Age"][train_df.Survived == 1].plot(kind='density', color='darkturquoise')
     train_df["Age"][train_df.Survived == 0].plot(kind='density', color='lightcoral')
     plt.legend(["Total",'Survived', 'Died'],fontsize=20)
     plt.xlim(-10,85)
     plt.grid()
     plt.xticks(np.arange(-10,90,10))
     plt.title("Age Distribution",fontsize=30)
     plt.xlabel("Age",fontsize=20)
     plt.ylabel("Density",fontsize=20)
     plt.show()
```



Umur penumpang kapal titanic didominasi oleh penumpang dengan umur 10 sampai 40 tahun

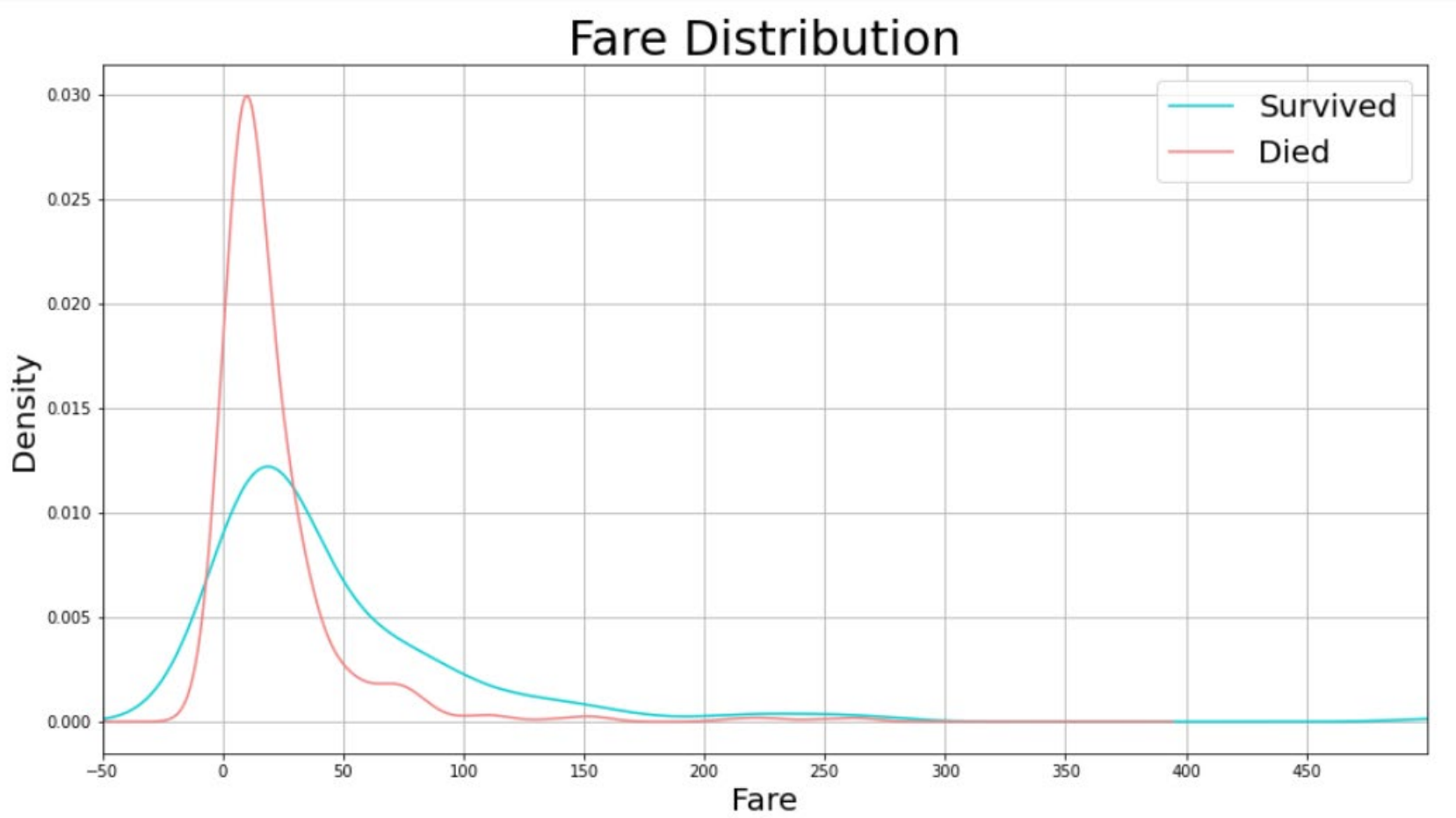
```
plt.figure(figsize=(15,8))
train_df["Fare"][train_df.Survived == 1].plot(kind='density', color='darkturquoise')
train_df["Fare"][train_df.Survived == 0].plot(kind='density', color='lightcoral')
plt.legend(['Survived', 'Died'],fontsize=20)
plt.xlim(--50,500)
plt.grid()
plt.xticks(np.arange(-50,500,50))
plt.title("Fare Distribution",fontsize=30)
plt.xlabel("Fare",fontsize=20)
plt.ylabel("Density",fontsize=20)
plt.show()
print("Tarif minimal", train_df["Fare"].min())
print("Tarif maksimal", train_df["Fare"].max())
```




```
Tarif minimal 0.0
Tarif maksimal 512.3292
```

Rentang tarif tiket yang dibayarkan penumpang bervariasi mulai dari 0 sampai 512

```
plt.figure(figsize=(15,8))
train_df["Fare"][train_df.Survived == 1].plot(kind='density', color='darkturquoise')
train_df["Fare"][train_df.Survived == 0].plot(kind='density', color='lightcoral')
plt.legend(['Survived', 'Died'],fontsize=20)
plt.xlim(--50,500)
plt.grid()
plt.xticks(np.arange(-50,500,50))
plt.title("Fare Distribution",fontsize=30)
plt.xlabel("Fare",fontsize=20)
plt.ylabel("Density",fontsize=20)
plt.show()
print("Tarif minimal", train_df["Fare"].min())
print("Tarif maksimal", train_df["Fare"].max())
```

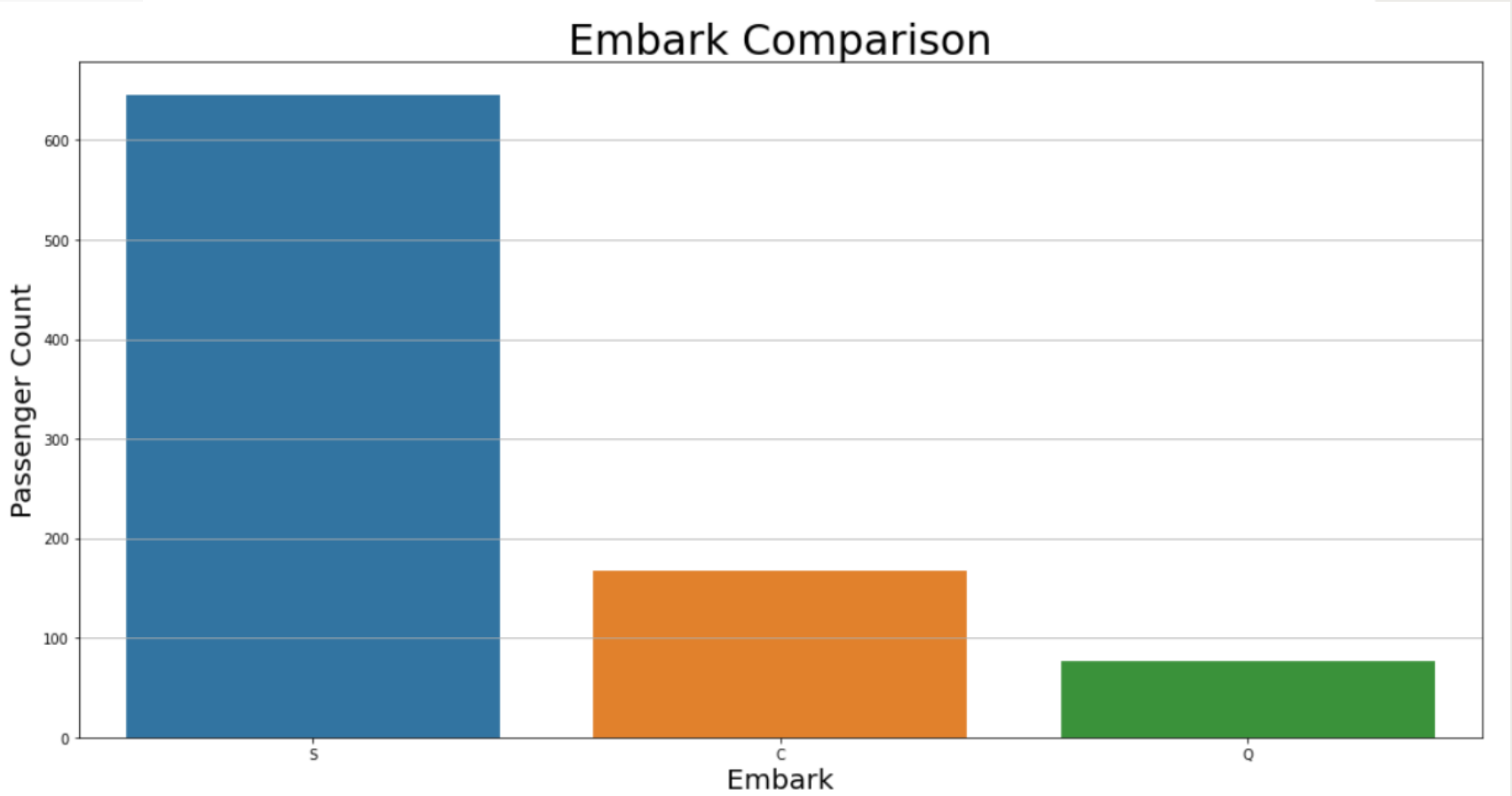



```
Tarif minimal 0.0
Tarif maksimal 512.3292
```

Rentang tarif tiket yang dibayarkan penumpang bervariasi mulai dari 0 sampai 512

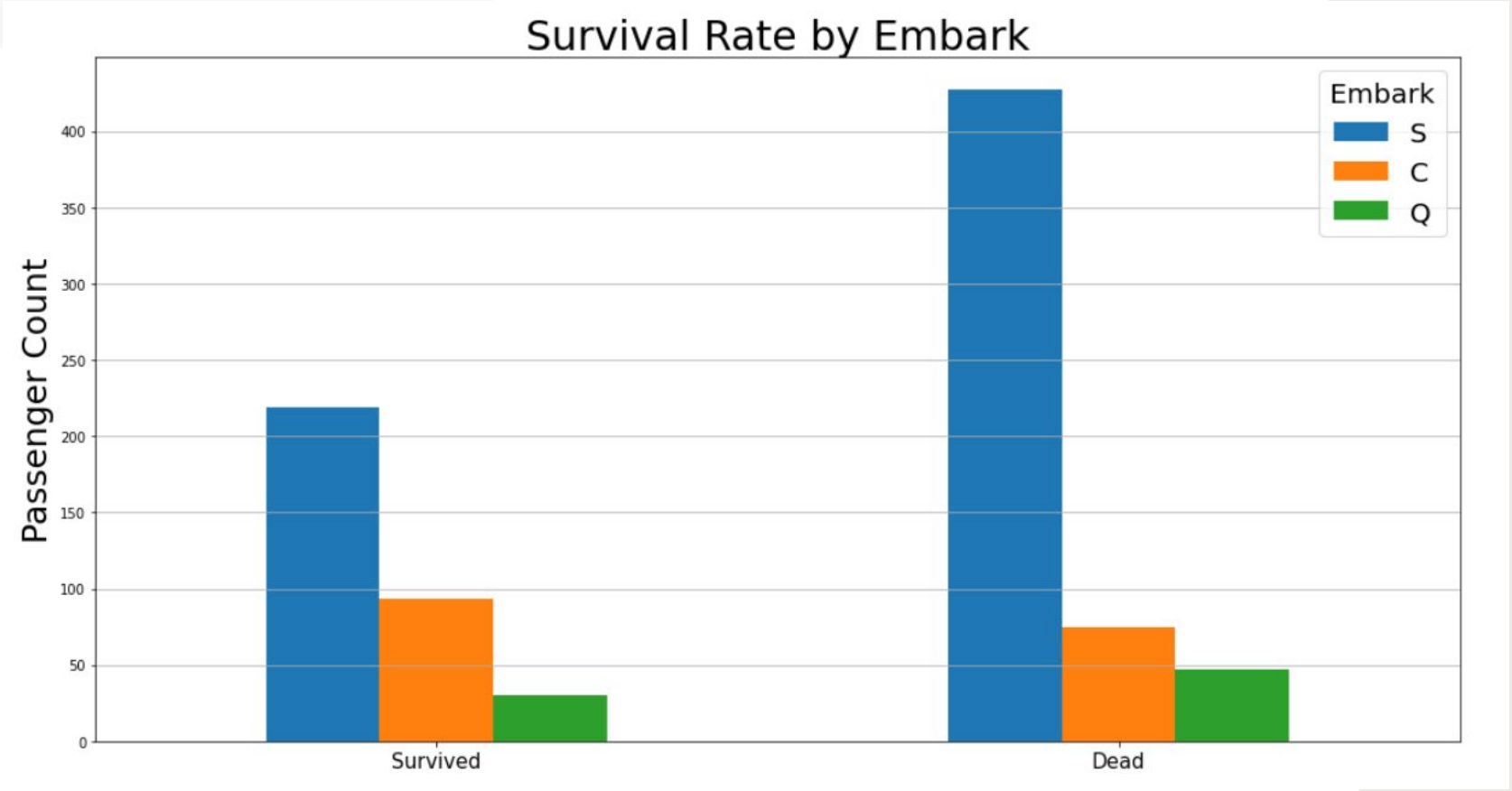
Perbandingan Embarked

```
[32] plt.figure(figsize=(15,8))
     sns.countplot(data=train_df, x='Embarked')
     plt.grid(axis='y')
     plt.xlabel('Embark',fontsize=20)
     plt.ylabel('Passenger Count',fontsize=20)
     plt.title("Embark Comparison",fontsize=30)
     plt.tight_layout()
```



Jumlah keberangkatan terbanyak berasal dari Southampton

```
bar_chart("Embarked")
plt.title("Survival Rate by Embark", fontsize=30)
plt.legend(title="Embark",title_fontsize=20,fontsize=20 )
plt.tight_layout()
```



Tempat keberangkatan dengan jumlah kematian terbanyak adalah di Southampton

Matriks Korelasi

```
[34] plt.figure(figsize=(15,8))
     sns.heatmap(train_df.corr(), annot=True, cmap="Blues")
     plt.show()
```

PENAMBAHAN FITUR

Fitur "FamilySize"

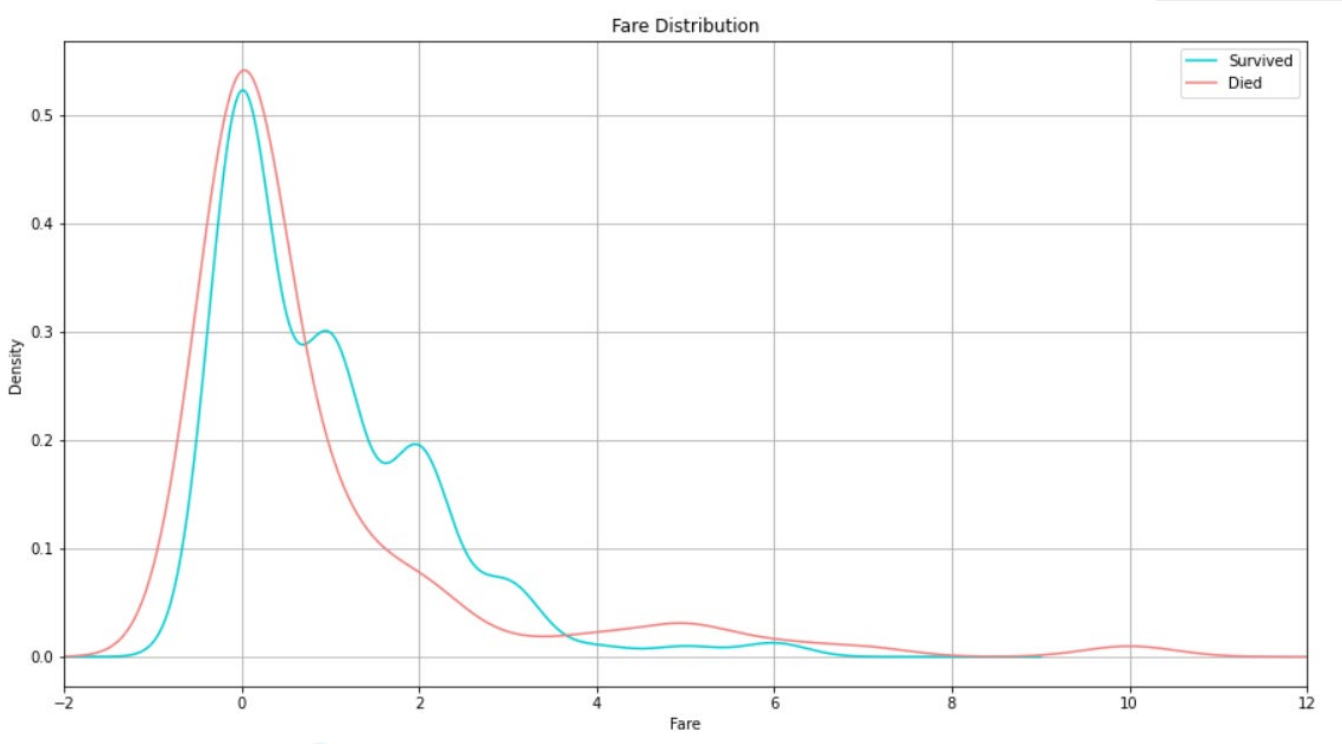
Karena fitur "SibSp" dan fitur "Parch" memiliki kesamaan yaitu sama-sama merepresentasikan jumlah keluarga, maka dibuatlah sebuah fitur baru yang merupakan gabungan fitur "SibSp" dan "Parch"

```
train_df['FamilySize'] = train_df['SibSp'] + train_df['Parch']

plt.figure(figsize=(15,8))
train_df["FamilySize"][train_df.Survived == 1].plot(kind='density', color='darkturquoise')
train_df["FamilySize"][train_df.Survived == 0].plot(kind='density', color='lightcoral')
plt.legend(['Survived', 'Died'])
plt.xlim(-2,12)
plt.grid()
plt.title("Fare Distribution")
plt.xlabel("Fare")
plt.show()

train_df[['FamilySize' , 'Survived']].groupby(['FamilySize'] , as_index=False).mean().sort_values(by='Survived' , ascending=False)
```

	FamilySize	Survived
3	3	0.724138
2	2	0.578431
1	1	0.552795
6	6	0.333333
0	0	0.303538
4	4	0.200000
5	5	0.136364
7	7	0.000000
8	10	0.000000

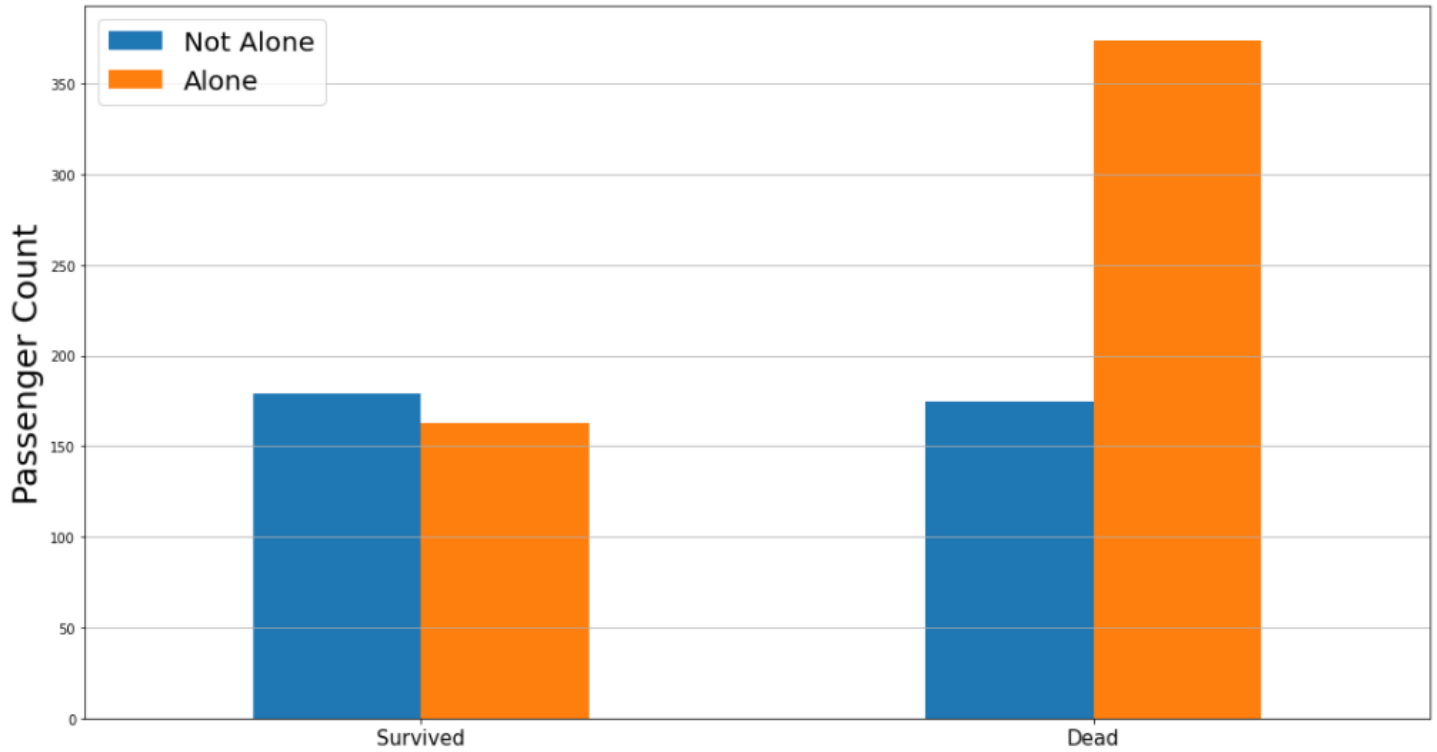


Fitur "IsAlone"

Selanjutnya ditambahkan fitur "IsAlone" yang merepresentasikan apakah seseorang tersebut sendirian atau bersama keluarganya saat menaiki kapal titanic

```
train_df['IsAlone'] = train_df['FamilySize'].apply(lambda x: 0 if x>0 else 1)

bar_chart("IsAlone")
plt.legend(['Not Alone', 'Alone'], fontsize=20)
plt.tight_layout()
```



Dapat dilihat bahwa jumlah orang yang naik ke kapal titanic sendirian lebih banyak tewas daripada yang naik dengan membawa keluarganya

Drop fitur "FamilySize", "SibSp", dan "Parch"

```
plt.figure(figsize=(15,8))
sns.heatmap(train_df.corr(), annot=True, cmap="Blues")
plt.show()
```

Karena fitur "FamilySize", "SibSp", "Parch", dan "IsAlone" saling berkaitan dan merepresentasikan data yang serupa maka hanya fitur "IsAlone" yang akan digunakan, dan selainnya akan di drop

```
train_df.drop(columns=['Parch' , 'FamilySize' , 'SibSp'] , axis=1 , inplace=True)
test_df.drop(columns=['Parch' , 'FamilySize' , 'SibSp'] , axis=1 , inplace=True)

train_df.head()
```

	PassengerId	Survived	Pclass	Name	Sex	Age	Ticket	Fare	Embarked	IsAlone
0	1	0	3	Braund, Mr. Owen Harris	male	22.0	A/5 21171	7.2500	S	0
1	2	1	1	Cumings, Mrs. John Bradley (Florence Briggs Th...	female	38.0	PC 17599	71.2833	C	0
2	3	1	3	Heikkinen, Miss. Laina	female	26.0	STON/O2. 3101282	7.9250	S	1
3	4	1	1	Futrelle, Mrs. Jacques Heath (Lily May Peel)	female	35.0	113803	53.1000	S	0
4	5	0	3	Allen, Mr. William Henry	male	35.0	373450	8.0500	S	1

PREPROCESSING

Drop fitur 'PassengerId', 'Name', dan 'Ticket'

Karena ketiga fitur tersebut tidak berpengaruh kepada target (Survived) maka ketiga fitur tersebut akan didrop

```
[40] train_df.drop(['PassengerId' , 'Name' , 'Ticket'] , axis=1 , inplace=True)
     test_df.drop(['PassengerId' , 'Name' , 'Ticket'] , axis=1 , inplace=True)
```

Pemisahan fitur dan target

```
from sklearn.linear_model import LogisticRegression
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler

y = train_df['Survived']
X = train_df.drop(['Survived'], axis=1)
```

MENGUBAH DATA KATEGORI MENJADI NUMERIK

```
X = pd.get_dummies(X , columns=['Sex' , 'Embarked'])
X.head()
```

	Pclass	Age	Fare	IsAlone	Sex_female	Sex_male	Embarked_C	Embarked_Q	Embarked_S
0	3	22.0	7.2500	0	0	1	0	0	1
1	1	38.0	71.2833	0	1	0	1	0	0
2	3	26.0	7.9250	1	1	0	0	0	1
3	1	35.0	53.1000	0	1	0	0	0	1
4	3	35.0	8.0500	1	0	1	0	0	1

```
test_df = pd.get_dummies(test_df , columns=['Sex' , 'Embarked'])
test_df.head()
```

	Pclass	Age	Fare	IsAlone	Sex_female	Sex_male	Embarked_C	Embarked_Q	Embarked_S
0	3	34.5	7.8292	1	0	1	0	1	0
1	3	47.0	7.0000	0	1	0	0	0	1
2	2	62.0	9.6875	1	0	1	0	1	0
3	3	27.0	8.6625	1	0	1	0	0	1
4	3	22.0	12.2875	0	1	0	0	0	1

Menormalisasi fitur "Age" dan "Fare"

agar memiliki bobot yang setara dengan fitur lainnya maka fitur "Age" dan "Fare" harus dinormalisasi terlebih dahulu

```
sc = StandardScaler()

X[['Age' , 'Fare']] = sc.fit_transform(X[['Age' , 'Fare']])
X.head()
```

	Pclass	Age	Fare	IsAlone	Sex_female	Sex_male	Embarked_C	Embarked_Q	Embarked_S
0	3	-0.565736	-0.502445	0	0	1	0	0	1
1	1	0.663861	0.786845	0	1	0	1	0	0
2	3	-0.258337	-0.488854	1	1	0	0	0	1
3	1	0.433312	0.420730	0	1	0	0	0	1
4	3	0.433312	-0.486337	1	0	1	0	0	1

```
test_df[['Age' , 'Fare']] = sc.fit_transform(test_df[['Age' , 'Fare']])
test_df.head()
```

	Pclass	Age	Fare	IsAlone	Sex_female	Sex_male	Embarked_C	Embarked_Q	Embarked_S
0	3	0.386231	-0.497413	1	0	1	0	1	0
1	3	1.371370	-0.512278	0	1	0	0	0	1
2	2	2.553537	-0.464100	1	0	1	0	1	0
3	3	-0.204852	-0.482475	1	0	1	0	0	1
4	3	-0.598908	-0.417492	0	1	0	0	0	1

Memisahkan data untuk training dan testing

```
[46] X_train, X_test, y_train, y_test = train_test_split(X,y, test_size=0.2, random_state=42)
```

Modelling

```
[47] from sklearn.ensemble import RandomForestClassifier
     from sklearn.metrics import accuracy_score

     random_forest = RandomForestClassifier(n_estimators=100).fit(X_train, y_train)

     y_pred = random_forest.predict(X_test)

     print('Accuracy:', accuracy_score(y_test, y_pred))
```

```
Accuracy: 0.8044692737430168
```

Prediksi untuk file submissi

```
[48] final_rf = RandomForestClassifier(n_estimators=110, max_depth= 8)

     final_rf.fit(X, y)
     final_pred = final_rf.predict(test_df)
```

```
[49] test_sub = pd.read_csv("/content/test.csv")

     submission = pd.DataFrame({
             "PassengerId": test_sub["PassengerId"],
             "Survived": final_pred
             })

     submission.to_csv('submission.csv', index=False)
```

AKURASI AKHIR

submission (1).csv 0.79425

5 days ago by Syahrul Apriansyah

add submission details