1. Import libraries
2. Import files
3. General view of data and data description
4. EDA

* Describe
* Missing values

1. Feature Analysis
2. Wanted column vs every other columns

* train[['Sex', 'Survived']].**groupby**('Sex', as\_index = False).mean()
* sns.**barplot**(x = 'Sex', y ='Survived', data = train);
* sns.**factorplot**(x = 'Pclass', y = 'Survived', hue = 'Sex', data = train, kind = 'bar');
* detect outliers (writing functions, 4 types of detect outliers)
* sns.**heatmap**(train[['Survived', 'SibSp', 'Parch', 'Age', 'Fare']].corr(), annot = True, fmt = '.2f', cmap = 'coolwarm');
* sns.**distplot**(train['Age'],);
* sns.**FacetGrid**(train, col = 'Survived');
* **map**(sns.distplot, 'Age');
* sns.**kdeplot**(train['Age'][train['Survived'] == 1], label = 'Survived');

1. Data Processing

* train = train.drop(['Cabin'], axis = 1)
* Missing values in training set train.isnull().sum().sort\_values(ascending = False)
* Compute the most frequent value of Embarked in training set

mode = train['Embarked'].dropna().mode()[0]

* Compute median

median = test['Fare'].dropna().median()

* Fill missing value with median

test['Fare'].fillna(median, inplace = True)

* + sns.distplot(combine['Fare'], label = 'Skewness: %.2f'%(combine['Fare'].skew()))

plt.legend(loc = 'best');

* + Apply log transformation to Fare column to reduce skewness

combine['Fare'] = combine['Fare'].map(lambda x: np.log(x) if x > 0 else 0)

### Like IsAlone, Title deduction should be made

1. Modelling

* Scikit-learn

#### Split training data

X\_train = train.drop('Survived', axis = 1)

Y\_train = train['Survived']

X\_test = test.drop('PassengerId', axis = 1).copy()

print("X\_train shape: ", X\_train.shape)

print("Y\_train shape: ", Y\_train.shape)

print("X\_test shape: ", X\_test.shape)

#### Random forest (many more)

random\_forest = RandomForestClassifier(n\_estimators = 100)

random\_forest.fit(X\_train, Y\_train)

Y\_pred = random\_forest.predict(X\_test)

acc\_random\_forest = round(random\_forest.score(X\_train, Y\_train) \* 100, 2)

acc\_random\_forest

1. Training Accuracy

* models = pd.DataFrame({'Model': ['Support Vector Machines', 'KNN', 'Logistic Regression',

'Random Forest', 'Naive Bayes', 'Perceptron', 'Stochastic Gradient Decent',

'Linear SVC', 'Decision Tree'],

'Score': [acc\_svc, acc\_knn, acc\_log, acc\_random\_forest, acc\_gaussian, acc\_perceptron,

acc\_sgd, acc\_linear\_svc, acc\_decision\_tree]})

models.sort\_values(by = 'Score', ascending = False, ignore\_index = True)

* K- Fold Cross validation

1. Preparing data for submission

* submit.to\_csv(r"C:\Users\udea3\OneDrive\Masaüstü\DS\aHaziran2022\titanic\_results.csv", index = False)