```
import pandas as pd
import numpy as np
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler
import tensorflow as tf
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import Dense, Dropout


train_data = pd.read_csv('train.csv')


X = train_data[['Pclass', 'Sex', 'Age', 'SibSp', 'Parch', 'Fare', 'Embarked']]


y = train_data['Survived']


X = pd.get_dummies(X, columns=['Sex', 'Embarked'], drop_first=True)


X['Age'].fillna(X['Age'].mean(), inplace=True)


X_train, X_val, y_train, y_val = train_test_split(X, y, test_size=0.2, random_state=42)


scaler = StandardScaler()
X_train[['Age', 'SibSp', 'Parch', 'Fare']] = scaler.fit_transform(X_train[['Age', 'SibSp', 'Parch', 'Fare']])
X_val[['Age', 'SibSp', 'Parch', 'Fare']] = scaler.transform(X_val[['Age', 'SibSp', 'Parch', 'Fare']])


model = Sequential()
model.add(Dense(64, activation='relu', input_dim=X_train.shape[1]))
model.add(Dropout(0.5))
model.add(Dense(32, activation='relu'))
model.add(Dense(1, activation='sigmoid'))


model.compile(loss='binary_crossentropy', optimizer='adam', metrics=['accuracy'])


model.fit(X_train, y_train, epochs=10, batch_size=32, validation_data=(X_val, y_val))
```

```
Epoch 1/10
23/23 [==============================] - 3s 16ms/step - loss: 0.7072 - accuracy: 0.5688 - val_loss: 0.6371 - val_accuracy: 0.5866
Epoch 2/10
23/23 [==============================] - 0s 3ms/step - loss: 0.6315 - accuracy: 0.6545 - val_loss: 0.6057 - val_accuracy: 0.6145
Epoch 3/10
23/23 [==============================] - 0s 4ms/step - loss: 0.6047 - accuracy: 0.6685 - val_loss: 0.5773 - val_accuracy: 0.6872
Epoch 4/10
23/23 [==============================] - 0s 4ms/step - loss: 0.5808 - accuracy: 0.7107 - val_loss: 0.5544 - val_accuracy: 0.6983
Epoch 5/10
23/23 [==============================] - 0s 4ms/step - loss: 0.5657 - accuracy: 0.7037 - val_loss: 0.5311 - val_accuracy: 0.7430
Epoch 6/10
23/23 [==============================] - 0s 4ms/step - loss: 0.5558 - accuracy: 0.7247 - val_loss: 0.5138 - val_accuracy: 0.7654
Epoch 7/10
23/23 [==============================] - 0s 4ms/step - loss: 0.5459 - accuracy: 0.7317 - val_loss: 0.4999 - val_accuracy: 0.7709
Epoch 8/10
23/23 [==============================] - 0s 3ms/step - loss: 0.5321 - accuracy: 0.7472 - val_loss: 0.4847 - val_accuracy: 0.8101
Epoch 9/10
23/23 [==============================] - 0s 4ms/step - loss: 0.5187 - accuracy: 0.7654 - val_loss: 0.4729 - val_accuracy: 0.8156
Epoch 10/10
23/23 [==============================] - 0s 4ms/step - loss: 0.5073 - accuracy: 0.7809 - val_loss: 0.4632 - val_accuracy: 0.8101
<keras.src.callbacks.History at 0x7fb6f0e69d50>
```

```
test_data = pd.read_csv('test.csv')


X_test = test_data[['Pclass', 'Sex', 'Age', 'SibSp', 'Parch', 'Fare', 'Embarked']]


X_test = pd.get_dummies(X_test, columns=['Sex', 'Embarked'], drop_first=True)


X_test['Age'].fillna(X_test['Age'].mean(), inplace=True)


X_test['Fare'].fillna(X_test['Fare'].mean(), inplace=True)


X_test[['Age', 'SibSp', 'Parch', 'Fare']] = scaler.transform(X_test[['Age', 'SibSp', 'Parch', 'Fare']])


predictions = model.predict(X_test)
```

```
14/14 [==============================] - 0s 2ms/step
```

```
predictions = (predictions > 0.5).astype(int)

submission = pd.DataFrame({'PassengerId': test_data['PassengerId'], 'Survived': predictions.flatten()})

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