Titanic Survival Analysis Project

In this project I will try and predict the survivors of the titanic test data set using linear regression. Additionally, I will identify which three variables explain survivability the most.

Importing

```
# Imports
import pandas as pd
import matplotlib.pyplot as plt
import numpy as np
import seaborn as sns
import statsmodels.formula.api as smf
from sklearn.metrics import r2_score
from sklearn.linear_model import LinearRegression
from sklearn.preprocessing import StandardScaler
```

Reading in the training and test data

```
train_data = pd.read_csv("https://raw.githubusercontent.com/austinkirwin/public-projects/refs/heads/main/Python_projects/Titanic_project/trai
test_data = pd.read_csv("http://raw.githubusercontent.com/austinkirwin/public-projects/refs/heads/main/Python_projects/Titanic_project/test.c
test_data.head()
```

	PassengerId	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare	Cabin	Embarked
0	892	3	Kelly, Mr. James	male	34.5	0	0	330911	7.8292	NaN	Q
1	893	3	Wilkes, Mrs. James (Ellen Needs)	female	47.0	1	0	363272	7.0000	NaN	S
2	894	2	Myles, Mr. Thomas Francis	male	62.0	0	0	240276	9.6875	NaN	Q
3	895	3	Wirz, Mr. Albert	male	27.0	0	0	315154	8.6625	NaN	S
4	896	3	Hirvonen, Mrs. Alexander (Helga E Lindqvist)	female	22.0	1	1	3101298	12.2875	NaN	S

Linear Models

I'm going to use a linear model to predict which passengers survive and which do not.

```
# Splitting the training data and removing NaN values
train_data = train_data.dropna()
# Features matrix
train_feature = train_data.drop(['Survived'], axis = 1)
# Target variable
train_target = train_data['Survived']
```

```
# Dropping unnecessary variables
train_feature = train_feature.drop(['Name','Ticket','Cabin','Embarked','Sex'], axis = 1)
```

```
# Compiling and fitting the full model
full_model = LinearRegression()
full_model.fit(train_feature, train_target)

prediction_values = full_model.predict(test_data.drop(['Cabin','Embarked','Name','Sex','Ticket'], axis = 1).dropna(),)
```

```
final_preds = pd.DataFrame(prediction_values)
final_preds.columns = ['Survived']
final_preds
```

	Survived
0	0.661221
1	0.599410
2	0.485545
3	0.727210
4	0.756698
...	...
326	1.035987
327	0.973622
328	0.833828
329	0.923213
330	0.743892

331 rows × 1 columns

```
# Reformatting each value to 'Yes' or 'No' for survival

final_preds[final_preds['Survived'] > .5] = 1
final_preds[final_preds['Survived'] < .5] = 0

map = {1: 'Yes', 0: 'No'}
final_preds['Survived'].map(map)
```

	Survived
0	Yes
1	Yes
2	No
3	Yes
4	Yes
...	...
326	Yes
327	Yes
328	Yes
329	Yes
330	Yes

331 rows × 1 columns

dtype: object

Implementing Decision Trees

For the purposes of learning, I will be using Tensorflow decision trees to try and predict the survivors.

```
!pip install tensorflow tensorflow_decision_forests
import tensorflow as tf
import tensorflow_decision_forests as tfdf
```

```
Requirement already satisfied: tensorflow in /usr/local/lib/python3.11/dist-packages (2.18.0)
Collecting tensorflow_decision_forests
  Downloading tensorflow_decision_forests-1.11.0-cp311-cp311-manylinux_2_17_x86_64.manylinux2014_x86_64.whl.metadata (6.0 kB)
Requirement already satisfied: absl-py>=1.0.0 in /usr/local/lib/python3.11/dist-packages (from tensorflow) (1.4.0)
Requirement already satisfied: astunparse>=1.6.0 in /usr/local/lib/python3.11/dist-packages (from tensorflow) (1.6.3)
Requirement already satisfied: flatbuffers>=24.3.25 in /usr/local/lib/python3.11/dist-packages (from tensorflow) (25.2.10)
Requirement already satisfied: gast!=0.5.0,!=0.5.1,!=0.5.2,>=0.2.1 in /usr/local/lib/python3.11/dist-packages (from tensorflow) (0.6.0
Requirement already satisfied: google-pasta>=0.1.1 in /usr/local/lib/python3.11/dist-packages (from tensorflow) (0.2.0)
Requirement already satisfied: libclang>=13.0.0 in /usr/local/lib/python3.11/dist-packages (from tensorflow) (18.1.1)
Requirement already satisfied: opt-einsum>=2.3.2 in /usr/local/lib/python3.11/dist-packages (from tensorflow) (3.4.0)
Requirement already satisfied: packaging in /usr/local/lib/python3.11/dist-packages (from tensorflow) (24.2)
Requirement already satisfied: protobuf!=4.21.0,!=4.21.1,!=4.21.2,!=4.21.3,!=4.21.4,!=4.21.5,<6.0.0dev,>=3.20.3 in /usr/local/lib/pyth
Requirement already satisfied: requests<3,>=2.21.0 in /usr/local/lib/python3.11/dist-packages (from tensorflow) (2.32.3)
Requirement already satisfied: setuptools in /usr/local/lib/python3.11/dist-packages (from tensorflow) (75.1.0)
Requirement already satisfied: six>=1.12.0 in /usr/local/lib/python3.11/dist-packages (from tensorflow) (1.17.0)
Requirement already satisfied: termcolor>=1.1.0 in /usr/local/lib/python3.11/dist-packages (from tensorflow) (2.5.0)
Requirement already satisfied: typing-extensions>=3.6.6 in /usr/local/lib/python3.11/dist-packages (from tensorflow) (4.12.2)
Requirement already satisfied: wrapt>=1.11.0 in /usr/local/lib/python3.11/dist-packages (from tensorflow) (1.17.2)
Requirement already satisfied: grpcio<2.0,>=1.24.3 in /usr/local/lib/python3.11/dist-packages (from tensorflow) (1.70.0)
Requirement already satisfied: tensorboard<2.19,>=2.18 in /usr/local/lib/python3.11/dist-packages (from tensorflow) (2.18.0)
Requirement already satisfied: keras>=3.5.0 in /usr/local/lib/python3.11/dist-packages (from tensorflow) (3.8.0)
Requirement already satisfied: numpy<2.1.0,>=1.26.0 in /usr/local/lib/python3.11/dist-packages (from tensorflow) (1.26.4)
Requirement already satisfied: h5py>=3.11.0 in /usr/local/lib/python3.11/dist-packages (from tensorflow) (3.12.1)
Requirement already satisfied: ml-dtypes<0.5.0,>=0.4.0 in /usr/local/lib/python3.11/dist-packages (from tensorflow) (0.4.1)
Requirement already satisfied: tensorflow-io-gcs-filesystem>=0.23.1 in /usr/local/lib/python3.11/dist-packages (from tensorflow) (0.37
Requirement already satisfied: pandas in /usr/local/lib/python3.11/dist-packages (from tensorflow_decision_forests) (2.2.2)
Requirement already satisfied: wheel in /usr/local/lib/python3.11/dist-packages (from tensorflow_decision_forests) (0.45.1)
Collecting wurlitzer (from tensorflow_decision_forests)
  Downloading wurlitzer-3.1.1-py3-none-any.whl.metadata (2.5 kB)
Requirement already satisfied: tf-keras~=2.17 in /usr/local/lib/python3.11/dist-packages (from tensorflow_decision_forests) (2.18.0)
Collecting ydf (from tensorflow_decision_forests)
  Downloading ydf-0.10.0-cp311-cp311-manylinux_2_17_x86_64.manylinux2014_x86_64.whl.metadata (3.5 kB)
Requirement already satisfied: rich in /usr/local/lib/python3.11/dist-packages (from keras>=3.5.0->tensorflow) (13.9.4)
Requirement already satisfied: namex in /usr/local/lib/python3.11/dist-packages (from keras>=3.5.0->tensorflow) (0.0.8)
Requirement already satisfied: optree in /usr/local/lib/python3.11/dist-packages (from keras>=3.5.0->tensorflow) (0.14.0)
Requirement already satisfied: charset-normalizer<4,>=2 in /usr/local/lib/python3.11/dist-packages (from requests<3,>=2.21.0->tensorfl
Requirement already satisfied: idna<4,>=2.5 in /usr/local/lib/python3.11/dist-packages (from requests<3,>=2.21.0->tensorflow) (3.10)
Requirement already satisfied: urllib3<3,>=1.21.1 in /usr/local/lib/python3.11/dist-packages (from requests<3,>=2.21.0->tensorflow) (2
Requirement already satisfied: certifi>=2017.4.17 in /usr/local/lib/python3.11/dist-packages (from requests<3,>=2.21.0->tensorflow) (2
Requirement already satisfied: markdown>=2.6.8 in /usr/local/lib/python3.11/dist-packages (from tensorboard<2.19,>=2.18->tensorflow) (
Requirement already satisfied: tensorboard-data-server<0.8.0,>=0.7.0 in /usr/local/lib/python3.11/dist-packages (from tensorboard<2.19
Requirement already satisfied: werkzeug>=1.0.1 in /usr/local/lib/python3.11/dist-packages (from tensorboard<2.19,>=2.18->tensorflow) (
Requirement already satisfied: python-dateutil>=2.8.2 in /usr/local/lib/python3.11/dist-packages (from pandas->tensorflow_decision_for
Requirement already satisfied: pytz>=2020.1 in /usr/local/lib/python3.11/dist-packages (from pandas->tensorflow_decision_forests) (202
Requirement already satisfied: tzdata>=2022.7 in /usr/local/lib/python3.11/dist-packages (from pandas->tensorflow_decision_forests) (2
Collecting protobuf!=4.21.0,!=4.21.1,!=4.21.2,!=4.21.3,!=4.21.4,!=4.21.5,<6.0.0dev,>=3.20.3 (from tensorflow)
  Downloading protobuf-5.29.3-cp38-abi3-manylinux2014_x86_64.whl.metadata (592 bytes)
Requirement already satisfied: MarkupSafe>=2.1.1 in /usr/local/lib/python3.11/dist-packages (from werkzeug>=1.0.1->tensorboard<2.19,>=
Requirement already satisfied: markdown-it-py>=2.2.0 in /usr/local/lib/python3.11/dist-packages (from rich->keras>=3.5.0->tensorflow)
Requirement already satisfied: pygments<3.0.0,>=2.13.0 in /usr/local/lib/python3.11/dist-packages (from rich->keras>=3.5.0->tensorflow
Requirement already satisfied: mdurl~=0.1 in /usr/local/lib/python3.11/dist-packages (from markdown-it-py>=2.2.0->rich->keras>=3.5.0->
Downloading tensorflow_decision_forests-1.11.0-cp311-cp311-manylinux_2_17_x86_64.manylinux2014_x86_64.whl (15.9 MB)
   ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 15.9/15.9 MB 77.2 MB/s eta 0:00:00
Downloading wurlitzer-3.1.1-py3-none-any.whl (8.6 kB)
Downloading ydf-0.10.0-cp311-cp311-manylinux_2_17_x86_64.manylinux2014_x86_64.whl (10.6 MB)
   ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 10.6/10.6 MB 92.8 MB/s eta 0:00:00
Downloading protobuf-5.29.3-cp38-abi3-manylinux2014_x86_64.whl (319 kB)
   ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 319.7/319.7 kB 26.1 MB/s eta 0:00:00
```

```
tfdf_train_data = pd.read_csv("https://raw.githubusercontent.com/austinkirwin/public-projects/refs/heads/main/Python_projects/Titanic_projec
tfdf_test_data = pd.read_csv("http://raw.githubusercontent.com/austinkirwin/public-projects/refs/heads/main/Python_projects/Titanic_project/
```

```
Uninstalling protobuf-4.25.6:
  Successfully uninstalled protobuf-4.25.6
Successfully installed protobuf-5.29.3 tensorflow_decision_forests-1.11.0 wurlitzer-3.1.1 ydf-0.10.0
```

Try YDF, the successor of TensorFlow Decision Forests using the same algorithms but with more features and faster training!

Tokenizing the names in the data and extracting any prefix

```
def preprocess(df):
  df = df.copy()

  def normalize_name(x):
    return " ".join([v.strip(",()[].\"'") for v in x.split(" ")])

  def ticket_number(x):
    return x.split(" ")[-1]

  def ticket_item(x):
    items = x.split(" ")
    if len(items) == 1:
      return "NONE"
    return "_".join(items[0:-1])

  df["Name"] = df["Name"].apply(normalize_name)
  df["Ticket_number"] = df["Ticket"].apply(ticket_number)
```

```
  df["Ticket_item"] = df["Ticket"].apply(ticket_item)
  return df

preprocessed_train_df = preprocess(tfdf_train_data)
preprocessed_test_df = preprocess(tfdf_test_data)

preprocessed_train_df.head(5)
```

	PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare	Cabin	Embarked	Ticket_number	Ticket_item
0	1	0	3	Braund Mr Owen Harris	male	22.0	1	0	A/5 21171	7.2500	NaN	S	21171	A/5
1	2	1	1	Cumings Mrs John Bradley Florence Briggs Thayer	female	38.0	1	0	PC 17599	71.2833	C85	C	17599	PC

We don't want to train the model on "PassengerID" and "Ticket" features.

```
input_features = list(preprocessed_train_df.columns)
input_features.remove("Ticket")
input_features.remove("PassengerId")
input_features.remove("Survived")
```

Converting dataset to TensorFlow dataset

```
def tokenize_names(features, labels = None):
  features["Name"] = tf.strings.split(features["Name"])
  return features, labels

train_ds = tfdf.keras.pd_dataframe_to_tf_dataset(preprocessed_train_df, label = "Survived").map(tokenize_names)
test_ds = tfdf.keras.pd_dataframe_to_tf_dataset(preprocessed_test_df).map(tokenize_names)
```

Training the model with default params

```
model = tfdf.keras.GradientBoostedTreesModel(
    verbose = 0,
    features = [tfdf.keras.FeatureUsage(name=n) for n in input_features],
    exclude_non_specified_features = True,
    random_seed = 10,
)
model.fit(train_ds)

self_evaluation = model.make_inspector().evaluation()
self_evaluation.loss
```

```
0.7801646590232849
```

Training model with improved default params

```
model2 = tfdf.keras.GradientBoostedTreesModel(
    verbose = 0,
    features=[tfdf.keras.FeatureUsage(name = n) for n in input_features],
    exclude_non_specified_features = True,
    min_examples = 1,
    categorical_algorithm = "RANDOM",
    shrinkage = 0.05,
    split_axis = "SPARSE_OBLIQUE",
```

```
    sparse_oblique_normalization = "MIN_MAX",
    sparse_oblique_num_projections_exponent=2.0,
    num_trees = 2000,
    random_seed = 10,
)

model2.fit(train_ds)

self_evaluation2 = model2.make_inspector().evaluation()
self_evaluation2.loss
```

```
0.8411012291908264
```

```
model2.summary()
```

```
Model: "gradient_boosted_trees_model_3"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
=================================================================
Total params: 1 (1.00 Byte)
Trainable params: 0 (0.00 Byte)
Non-trainable params: 1 (1.00 Byte)
_________________________________________________________________
Type: "GRADIENT_BOOSTED_TREES"
Task: CLASSIFICATION
Label: "__LABEL"

Input Features (11):
        Age
        Cabin
        Embarked
        Fare
        Name
        Parch
        Pclass
        Sex
        SibSp
        Ticket_item
        Ticket_number

No weights

Variable Importance: INV_MEAN_MIN_DEPTH:
    1.           "Sex"  0.459147 ################
    2.           "Age"  0.373636 ###########
    3.          "Fare"  0.263117 #####
    4.          "Name"  0.218394 ##
    5.   "Ticket_item"  0.181858
    6. "Ticket_number"  0.178881
    7.      "Embarked"  0.178076
    8.        "Pclass"  0.177209
    9.         "Parch"  0.176724
   10.         "SibSp"  0.171615

Variable Importance: NUM_AS_ROOT:
    1.  "Sex" 45.000000 ################
    2. "Name"  9.000000
    3.  "Age"  7.000000

Variable Importance: NUM_NODES:
    1.           "Age" 791.000000 ################
    2.          "Fare" 413.000000 ########
    3.          "Name" 124.000000 ##
    4.   "Ticket_item" 70.000000 #
    5.           "Sex" 63.000000 #
    6.         "Parch" 56.000000 #
    7.      "Embarked" 26.000000
    8. "Ticket_number" 21.000000
    9.        "Pclass" 13.000000
   10.         "SibSp"  6.000000

Variable Importance: SUM_SCORE:
```

Making predictions

```
def prediction_to_kaggle_format(model2, threshold=0.5):
    proba_survive = model2.predict(test_ds, verbose=0)[:, 0]  # Using model2 instead of model
    # Convert test_ds to a pandas DataFrame to access "PassengerId"
```

```
    test_df = next(iter(test_ds.batch(len(test_ds)))).as_numpy_iterator()
    test_df = pd.DataFrame(list(test_df))

    return pd.DataFrame({
        "PassengerId": test_df["PassengerId"],
        "Survived": (proba_survive >= threshold).astype(int)
    })

kaggle_predictions = prediction_to_kaggle_format(model2)
```

```
---------------------------------------------------------------------------
AttributeError                            Traceback (most recent call last)
<ipython-input-29-954743d19206> in <cell line: 0>()
     10     })
     11
---> 12 kaggle_predictions = prediction_to_kaggle_format(model2)

<ipython-input-29-954743d19206> in prediction_to_kaggle_format(model2, threshold)
      2     proba_survive = model2.predict(test_ds, verbose=0)[:, 0]  # Using model2 instead of model
      3     # Convert test_ds to a pandas DataFrame to access "PassengerId"
----> 4     test_df = next(iter(test_ds.batch(len(test_ds)))).as_numpy_iterator()
      5     test_df = pd.DataFrame(list(test_df))
      6

AttributeError: 'tuple' object has no attribute 'as_numpy_iterator'
```

Start coding or generate with AI.