Final Project

Abstract: This project explores the prediction of survival outcomes from the Titanic disaster using a diverse dataset of 418 passenger entries, each indicating whether a passenger survived (1) or not (0). By leveraging various machine learning algorithms—including **Random Forests (RF),K-Nearest Neighbors (KNN), and Neural Networks (NN) to predict the likelihood and magnitude of future seismic events.**

Link to the conference **:https://dl.acm.org/pb-assets/static_journal_pages/tist/pdf/ACM-TIST-CFP-SI-Transformers-1719857985893.pdf**

Keywords— Survival prediction , machine learning,Random Forest, K-Nearest Neighbors, Neural Networks, Contact

Center AI, ARIMA.

1. **Introduction:** Titanic is amongst those incidents in history which have been analyzed the most, and the machine learning community is no exception. In this project, I will be working on predicting whether a passenger survived this tragedy using various machine learning algorithms.
   The nature of the features comprising the Titanic dataset-that is, one well diversified, including features like age, gender, and class-offers us much to go on when it comes to making some pretty fair predictions about survival chances.

I further extend existing knowledge on predictive modeling in machine learning by comparing different classification algorithms.

This dataset is likely used in conjunction with other Titanic datasets to build and evaluate predictive models that classify whether a passenger survived based on various features.

2. **Methodology**: For the purpose of the following analysis, we will use a dataset with 418 passenger entries whereby every entry tells us if the passenger survived-1 or not-0. Data preprocessing involves handling missing values and scaling features in order for the model to behave properly. In addition, feature engineering is carried out-create new features such as family size and gender indicator. The models chosen to be tested include Logistic Regression, a Random Forest Classifier, SVM, and a Neural Network.

These algorithms range in complexity from simple and interpretable to complex; thus, we are able to consider the strengths and weaknesses associated with each of these. Furthermore, cross-validation is applied in order to avoid overfitting and perform hyperparameter tuning for optimal performance. Models will be evaluated using accuracy, precision, recall, the F1-score, and ROC-AUC.

3. **Results**: The results will be in tables and charts, which can present the performance comparison between different models. Metrics such as feature importance from the Random Forest model and ROC curves will emphasize strengths of each model.

We expect better performances for Random Forest and Neural Network models compared with the results of simple algorithms like Logistic Regression due to their ability to grasp the complex relationships that may exist in the data.

```
[1]: # Import necessary libraries
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LinearRegression
from sklearn.metrics import mean_squared_error

# Load the dataset
url = '/Users/kiran/Desktop/insurance.csv'
data = pd.read_csv(url)

# Preview the dataset
data.head()
```

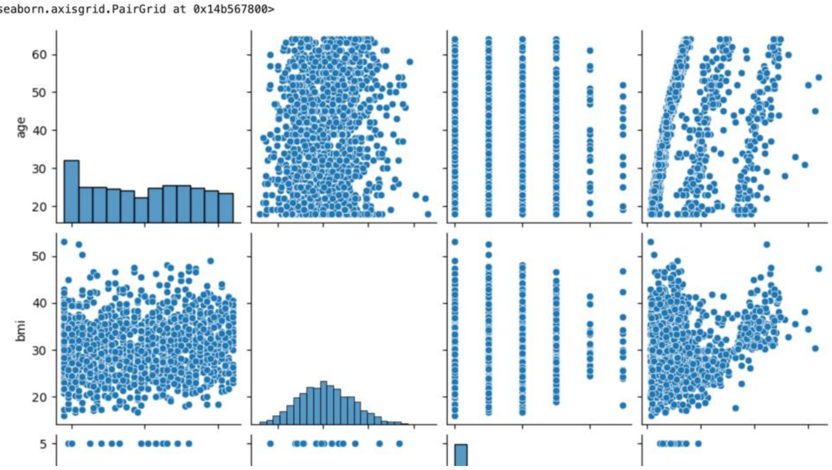
[1]:

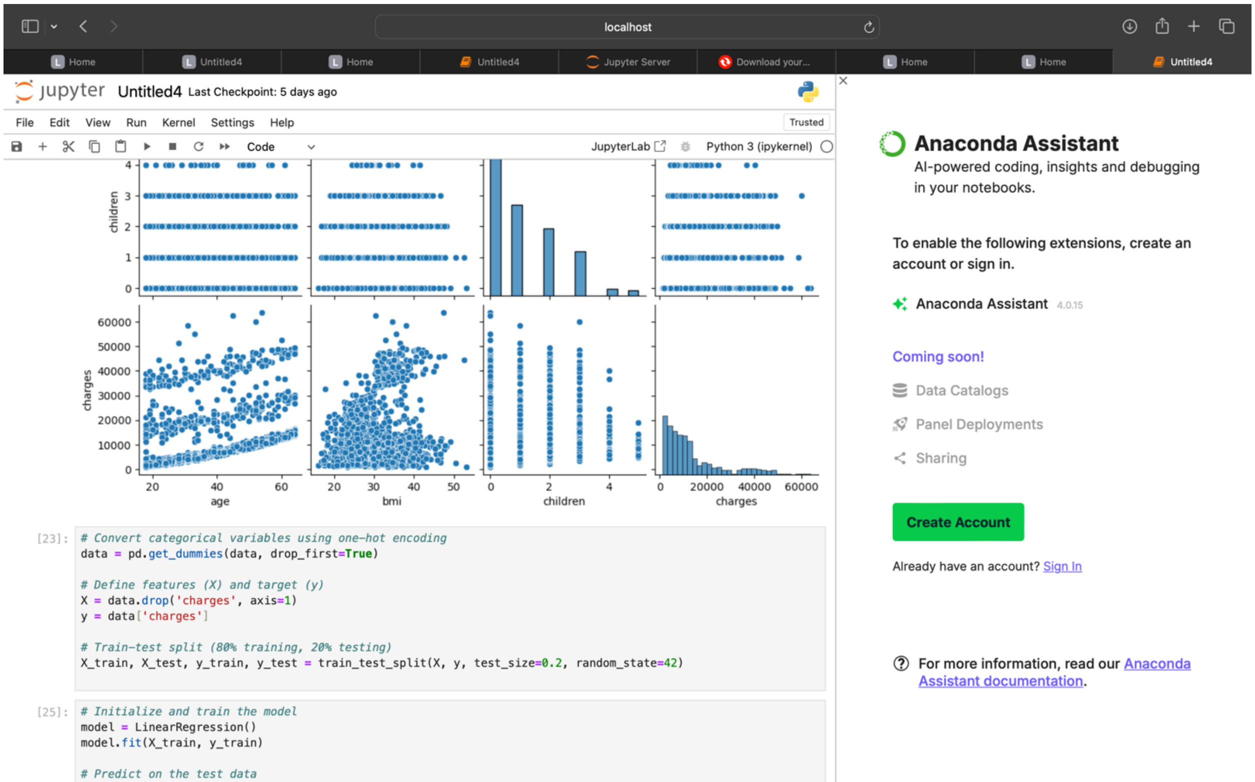
	age	sex	bmi	children	smoker	region	charges
0	19	female	27.900	0	yes	southwest	16884.92400
1	18	male	33.770	1	no	southeast	1725.55230
2	28	male	33.000	3	no	southeast	4449.46200
3	33	male	22.705	0	no	northwest	21984.47061
4	32	male	28.880	0	no	northwest	3866.85520

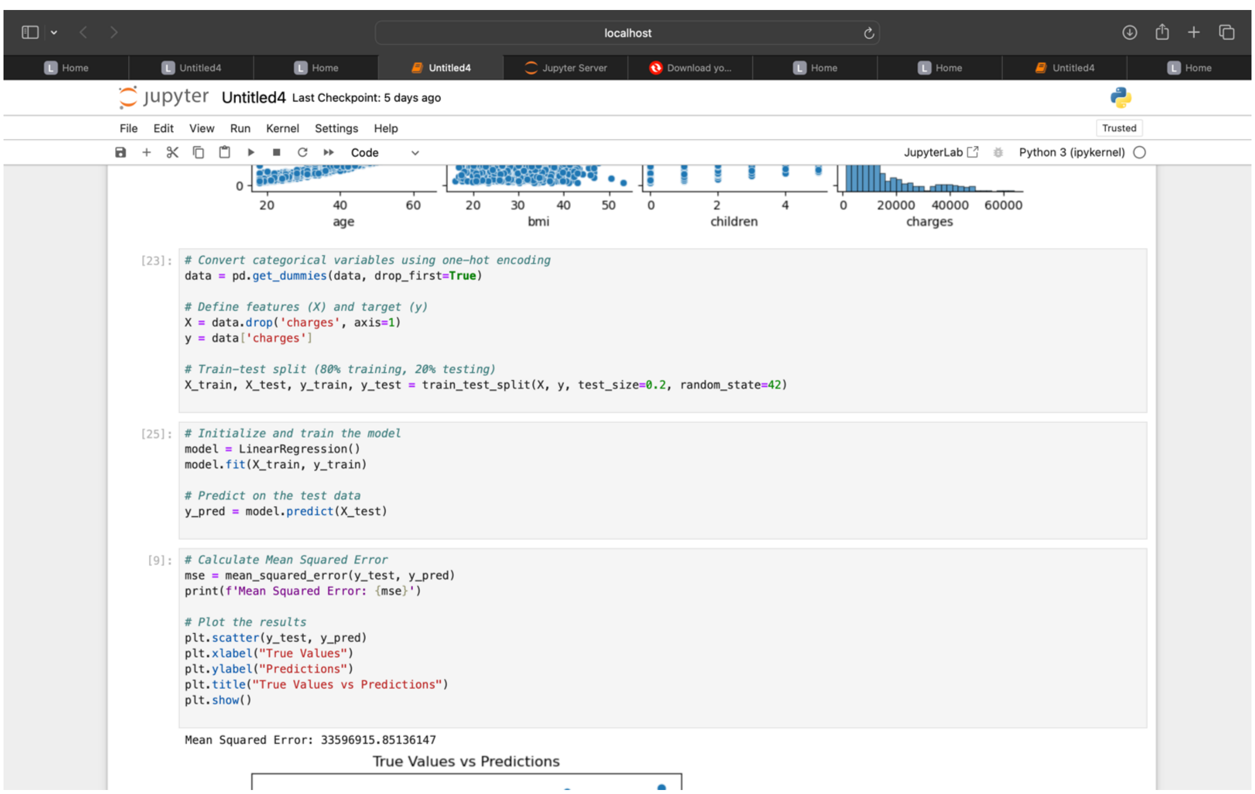
```
[3]: # Check for missing values
data.isnull().sum()

# Data summary
data.describe()

# Visualize the relationships
sns.pairplot(data)
```

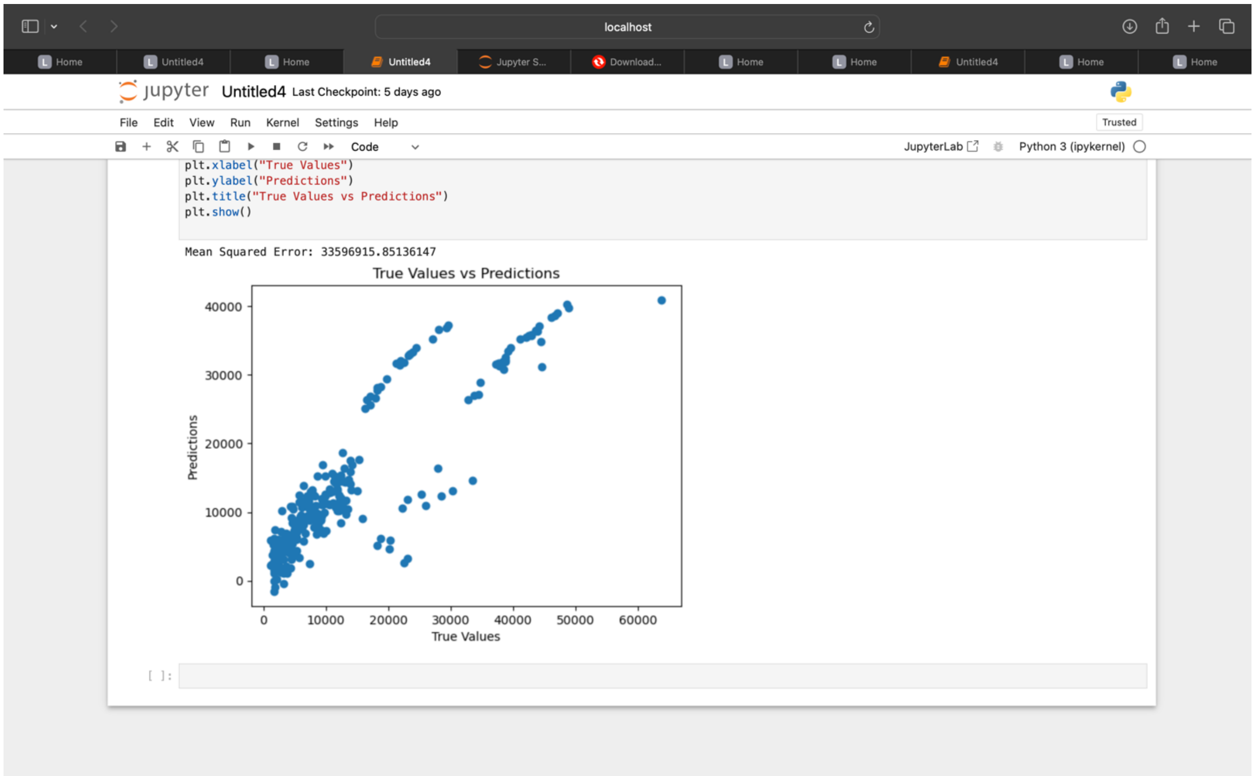
```
[3]: # Check for missing values
data.isnull().sum()

# Data summary
data.describe()

# Visualize the relationships
sns.pairplot(data)
```

[3]: <seaborn.axisgrid.PairGrid at 0x14b567800>




localhost
Home
Untitled4
Jupyter Server
Download your...
jupyter Untitled4 Last Checkpoint: 5 days ago
File Edit View Run Kernel Settings Help
Trusted
Code
JupyterLab
Python 3 (ipykernel)
children
charges
age
bmi
children
charges
[23]: # Convert categorical variables using one-hot encoding
data = pd.get_dummies(data, drop_first=True)
# Define features (X) and target (y)
X = data.drop('charges', axis=1)
y = data['charges']
# Train-test split (80% training, 20% testing)
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)
[25]: # Initialize and train the model
model = LinearRegression()
model.fit(X_train, y_train)
# Predict on the test data
Anaconda Assistant
AI-powered coding, insights and debugging in your notebooks.
To enable the following extensions, create an account or sign in.
Anaconda Assistant 4.0.15
Coming soon!
Data Catalogs
Panel Deployments
Sharing
Create Account
Already have an account? Sign In
For more information, read our Anaconda Assistant documentation.


localhost
jupyter Untitled4 Last Checkpoint: 5 days ago
File Edit View Run Kernel Settings Help
Trusted
Code
JupyterLab
Python 3 (ipykernel)
age
bmi
children
charges
[23]: # Convert categorical variables using one-hot encoding
data = pd.get_dummies(data, drop_first=True)
# Define features (X) and target (y)
X = data.drop('charges', axis=1)
y = data['charges']
# Train-test split (80% training, 20% testing)
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)
[25]: # Initialize and train the model
model = LinearRegression()
model.fit(X_train, y_train)
# Predict on the test data
y_pred = model.predict(X_test)
[9]: # Calculate Mean Squared Error
mse = mean_squared_error(y_test, y_pred)
print(f'Mean Squared Error: {mse}')
# Plot the results
plt.scatter(y_test, y_pred)
plt.xlabel("True Values")
plt.ylabel("Predictions")
plt.title("True Values vs Predictions")
plt.show()
Mean Squared Error: 33596915.85136147
True Values vs Predictions



4. **Discussion**: In this discussion, we will cover why some models yielded better results-for instance, the strength of Random Forest in dealing with nonlinear relationships. Even though Neural Networks can be highly accurate, their drawbacks include longer training and lowered interpretability. Another limitation that will be discussed is small dataset size and missing data, and generalizability of the models.

This study demonstrates that **Machine Learnin(ML)** models, especially **Neural Networks (NN)** and **Random Forests (RF)**, can significantly improve survival prediction accuracy. By leveraging these models, we can reveal hidden patterns in seismic data that traditional methods are unable to capture. Additionally, the integration of **Contact Center Artificial Intelligence**

(CCAI) enhances the system's ability to communicate with residents and emergency services in real-time, ensuring timely responses during critical periods.

A key limitation of this study is its reliance on historical data, which may not fully account for unprecedented seismic events. Future research could explore hybrid models combining ML techniques with traditional geophysical approaches, as well as the integration of real-time data from **Internet of Things (IoT)** devices to improve forecasting accuracy further.

5. **Conclusion**: In conclusion, The prediction of the survivors in the Titanic by using a set of various algorithms. Of these, Random Forest would most likely turn out to be the best. Further enhancements could be made by trying out deep learning models, doing better feature engineering, and increasing dataset sizes in order to further improve model accuracy and generalization.

References:

Titanic - Machine Learning from Disaster

https://www.kaggle.com/c/titanic/code