36106_25AU-AT1_25589351_experiment_3

March 29, 2025

1 Experiment Notebook

1.1 0. Setup Environment

1.1.1 0.a Install Mandatory Packages

Do not modify this code before running it

[1]:
```
# Do not modify this code

import os
import sys
from pathlib import Path

COURSE = "36106"
ASSIGNMENT = "AT1"
DATA = "data"

asgmt_path = f"{COURSE}/assignment/{ASSIGNMENT}"
root_path = "./"

print("###### Install required Python packages ######")
! pip install -r https://raw.githubusercontent.com/aso-uts/labs_datasets/main/
↪36106-mlaa/requirements.txt

if os.getenv("COLAB_RELEASE_TAG"):

    from google.colab import drive
    from pathlib import Path

    print("\n###### Connect to personal Google Drive ######")
    gdrive_path = "/content/gdrive"
    drive.mount(gdrive_path)
    root_path = f"{gdrive_path}/MyDrive/"

print("\n###### Setting up folders ######")
folder_path = Path(f"{root_path}/{asgmt_path}/") / DATA
```

```
folder_path.mkdir(parents=True, exist_ok=True)
print(f"\nYou can now save your data files in: {folder_path}")

if os.getenv("COLAB_RELEASE_TAG"):
    %cd {folder_path}
```

```
###### Install required Python packages ######
Requirement already satisfied: pandas==2.2.2 in /usr/local/lib/python3.11/dist-
packages (from -r https://raw.githubusercontent.com/aso-
uts/labs_datasets/main/36106-mlaa/requirements.txt (line 1)) (2.2.2)
Requirement already satisfied: scikit-learn==1.6.1 in
/usr/local/lib/python3.11/dist-packages (from -r
https://raw.githubusercontent.com/aso-
uts/labs_datasets/main/36106-mlaa/requirements.txt (line 2)) (1.6.1)
Requirement already satisfied: altair==5.5.0 in /usr/local/lib/python3.11/dist-
packages (from -r https://raw.githubusercontent.com/aso-
uts/labs_datasets/main/36106-mlaa/requirements.txt (line 3)) (5.5.0)
Requirement already satisfied: numpy>=1.23.2 in /usr/local/lib/python3.11/dist-
packages (from pandas==2.2.2->-r https://raw.githubusercontent.com/aso-
uts/labs_datasets/main/36106-mlaa/requirements.txt (line 1)) (2.0.2)
Requirement already satisfied: python-dateutil>=2.8.2 in
/usr/local/lib/python3.11/dist-packages (from pandas==2.2.2->-r
https://raw.githubusercontent.com/aso-
uts/labs_datasets/main/36106-mlaa/requirements.txt (line 1)) (2.8.2)
Requirement already satisfied: pytz>=2020.1 in /usr/local/lib/python3.11/dist-
packages (from pandas==2.2.2->-r https://raw.githubusercontent.com/aso-
uts/labs_datasets/main/36106-mlaa/requirements.txt (line 1)) (2025.1)
Requirement already satisfied: tzdata>=2022.7 in /usr/local/lib/python3.11/dist-
packages (from pandas==2.2.2->-r https://raw.githubusercontent.com/aso-
uts/labs_datasets/main/36106-mlaa/requirements.txt (line 1)) (2025.1)
Requirement already satisfied: scipy>=1.6.0 in /usr/local/lib/python3.11/dist-
packages (from scikit-learn==1.6.1->-r https://raw.githubusercontent.com/aso-
uts/labs_datasets/main/36106-mlaa/requirements.txt (line 2)) (1.14.1)
Requirement already satisfied: joblib>=1.2.0 in /usr/local/lib/python3.11/dist-
packages (from scikit-learn==1.6.1->-r https://raw.githubusercontent.com/aso-
uts/labs_datasets/main/36106-mlaa/requirements.txt (line 2)) (1.4.2)
Requirement already satisfied: threadpoolctl>=3.1.0 in
/usr/local/lib/python3.11/dist-packages (from scikit-learn==1.6.1->-r
https://raw.githubusercontent.com/aso-
uts/labs_datasets/main/36106-mlaa/requirements.txt (line 2)) (3.6.0)
Requirement already satisfied: jinja2 in /usr/local/lib/python3.11/dist-packages
(from altair==5.5.0->-r https://raw.githubusercontent.com/aso-
uts/labs_datasets/main/36106-mlaa/requirements.txt (line 3)) (3.1.6)
Requirement already satisfied: jsonschema>=3.0 in
/usr/local/lib/python3.11/dist-packages (from altair==5.5.0->-r
https://raw.githubusercontent.com/aso-
uts/labs_datasets/main/36106-mlaa/requirements.txt (line 3)) (4.23.0)
```

```
Requirement already satisfied: narwhals>=1.14.2 in
/usr/local/lib/python3.11/dist-packages (from altair==5.5.0->-r
https://raw.githubusercontent.com/aso-
uts/labs_datasets/main/36106-mlaa/requirements.txt (line 3)) (1.31.0)
Requirement already satisfied: packaging in /usr/local/lib/python3.11/dist-
packages (from altair==5.5.0->-r https://raw.githubusercontent.com/aso-
uts/labs_datasets/main/36106-mlaa/requirements.txt (line 3)) (24.2)
Requirement already satisfied: typing-extensions>=4.10.0 in
/usr/local/lib/python3.11/dist-packages (from altair==5.5.0->-r
https://raw.githubusercontent.com/aso-
uts/labs_datasets/main/36106-mlaa/requirements.txt (line 3)) (4.12.2)
Requirement already satisfied: attrs>=22.2.0 in /usr/local/lib/python3.11/dist-
packages (from jsonschema>=3.0->altair==5.5.0->-r
https://raw.githubusercontent.com/aso-
uts/labs_datasets/main/36106-mlaa/requirements.txt (line 3)) (25.3.0)
Requirement already satisfied: jsonschema-specifications>=2023.03.6 in
/usr/local/lib/python3.11/dist-packages (from jsonschema>=3.0->altair==5.5.0->-r
https://raw.githubusercontent.com/aso-
uts/labs_datasets/main/36106-mlaa/requirements.txt (line 3)) (2024.10.1)
Requirement already satisfied: referencing>=0.28.4 in
/usr/local/lib/python3.11/dist-packages (from jsonschema>=3.0->altair==5.5.0->-r
https://raw.githubusercontent.com/aso-
uts/labs_datasets/main/36106-mlaa/requirements.txt (line 3)) (0.36.2)
Requirement already satisfied: rpds-py>=0.7.1 in /usr/local/lib/python3.11/dist-
packages (from jsonschema>=3.0->altair==5.5.0->-r
https://raw.githubusercontent.com/aso-
uts/labs_datasets/main/36106-mlaa/requirements.txt (line 3)) (0.23.1)
Requirement already satisfied: six>=1.5 in /usr/local/lib/python3.11/dist-
packages (from python-dateutil>=2.8.2->pandas==2.2.2->-r
https://raw.githubusercontent.com/aso-
uts/labs_datasets/main/36106-mlaa/requirements.txt (line 1)) (1.17.0)
Requirement already satisfied: MarkupSafe>=2.0 in
/usr/local/lib/python3.11/dist-packages (from jinja2->altair==5.5.0->-r
https://raw.githubusercontent.com/aso-
uts/labs_datasets/main/36106-mlaa/requirements.txt (line 3)) (3.0.2)

###### Connect to personal Google Drive ######
Mounted at /content/gdrive

###### Setting up folders ######

You can now save your data files in:
/content/gdrive/MyDrive/36106/assignment/AT1/data
/content/gdrive/MyDrive/36106/assignment/AT1/data
```

1.1.2 0.b Disable Warnings Messages

Do not modify this code before running it

```
[ ]: import warnings
     warnings.simplefilter(action='ignore', category=FutureWarning)
```

1.1.3 0.c Install Additional Packages

If you are using additional packages, you need to install them here using the command: `! pip install <package_name>`

```
[ ]: # <Student to fill this section>
```

1.1.4 0.d Import Packages

```
[1]: import ipywidgets as widgets
     import pandas as pd
     import altair as alt
     import numpy as np
     from sklearn.metrics import mean_squared_error, r2_score, mean_absolute_error
     from itertools import product
```

1.2 A. Project Description

```
[2]: # @title Student Information
     wgt_student_name = widgets.Text(
         value="Fatemeh Elyasifar",
         placeholder='<student to fill this section>',
         description='Student Name:',
         style={'description_width': 'initial'},
         disabled=False
     )

     wgt_student_id = widgets.Text(
         value="25589351",
         placeholder='<student to fill this section>',
         description='Student Id:',
         style={'description_width': 'initial'},
         disabled=False
     )

     widgets.HBox([wgt_student_name, wgt_student_id])
```

```
HBox(children=(Text(value='Fatemeh Elyasifar', description='Student Name:',␣
↪placeholder='<student to fill this…
```

```
[3]: print("Student Name:", wgt_student_name.value)
     print("Student Id:", wgt_student_id.value)
```

Student Name: Fatemeh Elyasifar
Student Id: 25589351

[4]:
```
# @title Experiment ID

wgt_experiment_id = widgets.BoundedIntText(
    value="3",
    min=0,
    max=3,
    step=1,
    description='Experiment ID:',
    style={'description_width': 'initial'},
    disabled=False
)
wgt_experiment_id
```

```
BoundedIntText(value=3, description='Experiment ID:', max=3,␣
↪style=DescriptionStyle(description_width='initial…
```

[5]:
```
print("Experiment ID:", wgt_experiment_id.value)
```

```
Experiment ID: 3
```

[6]:
```
# @title Business Objective

wgt_business_objective = widgets.Textarea(
    value="The main objective is to develop a machine learning model that␣
↪accurately predicts rental prices specifically for affordable properties in␣
↪Australia, excluding luxury homes. This model aims to assist real estate␣
↪agencies, property investors, and tenants in making informed decisions based␣
↪on the features of affordable housing and market trends. The key success␣
↪metric is RMSE, with the goal of achieving an RMSE score of less than 16 on␣
↪the validation set, which quantifies the prediction error. Additionally,␣
↪feature tuning will be necessary to optimise the model's performance and␣
↪ensure accurate predictions based on the most relevant property␣
↪characteristics.",
    placeholder='<student to fill this section>',
    description='Business Objective:',
    disabled=False,
    style={'description_width': 'initial'},
    layout=widgets.Layout(height="100%", width="auto")
)
wgt_business_objective
```

```
Textarea(value='The main objective is to develop a machine learning model that␣
↪accurately predicts rental pric…
```

[7]:
```
print("Business Objective:", wgt_business_objective.value)
```

```
Business Objective: The main objective is to develop a machine learning model
that accurately predicts rental prices specifically for affordable properties in
Australia, excluding luxury homes. This model aims to assist real estate
agencies, property investors, and tenants in making informed decisions based on
the features of affordable housing and market trends. The key success metric is
RMSE, with the goal of achieving an RMSE score of less than 16 on the validation
set, which quantifies the prediction error. Additionally, feature tuning will be
necessary to optimise the model’s performance and ensure accurate predictions
based on the most relevant property characteristics.
```

1.3 B. Experiment Description

[8]:
```python
# @title Experiment Hypothesis

wgt_experiment_hypothesis = widgets.Textarea(
    value="Using a non-parametric regression approach like K-Nearest Neighbors
 ↪(KNN) can provide accurate and reliable rental price predictions based on
 ↪key property features such as location, size, and amenities. By leveraging
 ↪machine learning, businesses can optimise pricing strategies, enhance
 ↪customer decision-making, and improve profitability in the real estate
 ↪sector. Null Hypothesis (H ): Adjusting the number of neighbors (n_neighbors)
 ↪and the distance metric (p) does not significantly impact the RMSE of the
 ↪KNN regression model.",
    placeholder='<student to fill this section>',
    description='Experiment Hypothesis:',
    disabled=False,
    style={'description_width': 'initial'},
    layout=widgets.Layout(height="100%", width="auto")
)
wgt_experiment_hypothesis
```

```
Textarea(value='Using a non-parametric regression approach like K-Nearest
 ↪Neighbors (KNN) can provide accurate…
```

[9]:
```python
print("Experiment Hypothesis:", wgt_experiment_hypothesis.value)
```

```
Experiment Hypothesis: Using a non-parametric regression approach like K-Nearest
Neighbors (KNN) can provide accurate and reliable rental price predictions based
on key property features such as location, size, and amenities. By leveraging
machine learning, businesses can optimise pricing strategies, enhance customer
decision-making, and improve profitability in the real estate sector. Null
Hypothesis (H ): Adjusting the number of neighbors (n_neighbors) and the
distance metric (p) does not significantly impact the RMSE of the KNN regression
model.
```

[10]:
```
# @title Experiment Expectations

wgt_experiment_expectations = widgets.Textarea(
    value="Train multiple KNN regression models with different n_neighbors and␣
 ↪p values. Compare RMSE scores to determine the best settings for capturing␣
 ↪local rental price patterns. Evaluate computational efficiency and␣
 ↪scalability of KNN regression. Provide insights into whether KNN is a␣
 ↪suitable model or if further feature engineering is required.",
    placeholder='<student to fill this section>',
    description='Experiment Expectations:',
    disabled=False,
    style={'description_width': 'initial'},
    layout=widgets.Layout(height="100%", width="auto")
)
wgt_experiment_expectations
```

```
Textarea(value='Train multiple KNN regression models with different n_neighbors␣
 ↪and p values. Compare RMSE sco…
```

[11]:
```
print("Experiment Expectations:", wgt_experiment_expectations.value)
```

```
Experiment Expectations: Train multiple KNN regression models with different
n_neighbors and p values. Compare RMSE scores to determine the best settings for
capturing local rental price patterns. Evaluate computational efficiency and
scalability of KNN regression. Provide insights into whether KNN is a suitable
model or if further feature engineering is required.
```

1.4 C. Data Understanding

1.4.1 C.1 Load Datasets

Do not change this code

[3]:
```
# Load training data
X_train = pd.read_csv(folder_path / 'X_train.csv')
y_train = pd.read_csv(folder_path / 'y_train.csv')
```

[4]:
```
# Load validation data
X_val = pd.read_csv(folder_path / 'X_val.csv')
y_val = pd.read_csv(folder_path / 'y_val.csv')
```

[5]:
```
# Load testing data
X_test = pd.read_csv(folder_path / 'X_test.csv')
y_test = pd.read_csv(folder_path / 'y_test.csv')
```

1.5 D. Feature Selection

[ ]:
```
# <Student to fill this section>

features_list =␣
↪['number_of_bedrooms','floor_area','number_of_bathrooms','month','level_numerator','level_r
↪Family','tenancy_preference_Family','suburb_Brisbane','suburb_Canberra','suburb_Melbourne',
```

[12]:
```
# @title Feature Selection Explanation

wgt_feat_selection_explanation = widgets.Textarea(
    value="These attributes give a clear view of the property's features and␣
↪market conditions. They focus on the main factors that affect rental rates,␣
↪helping to understand what influences pricing and demand.",
    placeholder='<student to fill this section>',
    description='Feature Selection Explanation:',
    disabled=False,
    style={'description_width': 'initial'},
    layout=widgets.Layout(height="100%", width="auto")
)
wgt_feat_selection_explanation
```

```
Textarea(value="These attributes give a clear view of the property's features␣
↪and market conditions. They focu…
```

[14]:
```
print('Feature Selection Explanation:', wgt_feat_selection_explanation.value)
```

```
Feature Selection Explanation: These attributes give a clear view of the
property's features and market conditions. They focus on the main factors that
affect rental rates, helping to understand what influences pricing and demand.
```

1.6 E. Train Machine Learning Model

1.6.1 E.1 Import Algorithm

Provide some explanations on why you believe this algorithm is a good fit

[6]:
```
# <Student to fill this section>
from sklearn.neighbors import KNeighborsRegressor
```

[15]:
```
# @title Algorithm Selection Explanation

wgt_algo_selection_explanation = widgets.Textarea(
```

```
    value="K-Nearest Neighbors (KNN) Regression is chosen because it is a␣
↪non-parametric, instance-based learning algorithm that makes predictions␣
↪based on the average of the k nearest neighbors. It is well-suited for␣
↪problems where the relationship between features and target values is␣
↪non-linear and where local patterns in the data matter.",
    placeholder='<student to fill this section>',
    description='Algorithm Selection Explanation:',
    disabled=False,
    style={'description_width': 'initial'},
    layout=widgets.Layout(height="100%", width="auto")
)
wgt_algo_selection_explanation
```

```
Textarea(value='K-Nearest Neighbors (KNN) Regression is chosen because it is a␣
↪non-parametric, instance-based …
```

[16]:
```
print('Algorithm Selection Explanation:', wgt_algo_selection_explanation.value)
```

```
Algorithm Selection Explanation: K-Nearest Neighbors (KNN) Regression is chosen
because it is a non-parametric, instance-based learning algorithm that makes
predictions based on the average of the k nearest neighbors. It is well-suited
for problems where the relationship between features and target values is non-
linear and where local patterns in the data matter.
```

1.6.2 E.2 Set Hyperparameters

Provide some explanations on why you believe this algorithm is a good fit

[7]:
```
# <Student to fill this section>
n_neighbors_list = [1, 2, 3, 5, 7, 9, 11]
p_list = [1, 2]
```

[17]:
```
# @title Hyperparameters Selection Explanation

wgt_hyperparams_selection_explanation = widgets.Textarea(
    value="Values in n_neighbors_list allow to explore a range of neighborhood␣
↪sizes. Smaller values capture local patterns, while larger values provide␣
↪more generalised predictions. The p parameter in KNN regression defines the␣
↪distance metric used to calculate the distance between data points. p = 1:␣
↪Uses Manhattan distance and p = 2: Uses Euclidean distance.",
    placeholder='<student to fill this section>',
    description='Hyperparameters Selection Explanation:',
    disabled=False,
    style={'description_width': 'initial'},
    layout=widgets.Layout(height="100%", width="auto")
)
wgt_hyperparams_selection_explanation
```

```
Textarea(value='Values in n_neighbors_list allow to explore a range of␣
↪neighborhood sizes. Smaller values capt…
```

[18]:
```python
print('Hyperparameters Selection Explanation:',␣
↪wgt_hyperparams_selection_explanation.value)
```

```
Hyperparameters Selection Explanation: Values in n_neighbors_list allow to
explore a range of neighborhood sizes. Smaller values capture local patterns,
while larger values provide more generalised predictions. The p parameter in KNN
regression defines the distance metric used to calculate the distance between
data points. p = 1: Uses Manhattan distance and p = 2: Uses Euclidean distance.
```

1.6.3 E.3 Fit Model

[9]:
```python
# <Student to fill this section>
predictions = {}

for n_neighbors, p in product(n_neighbors_list, p_list):
    model = KNeighborsRegressor(n_neighbors=n_neighbors, p=p)
    model.fit(X_train, y_train)

    y_pred = model.predict(X_val)

    # Store predictions in a dictionary
    key = f"y_pred_{n_neighbors}_{p}"
    predictions[key] = y_pred
```

1.6.4 E.4 Model Technical Performance

Provide some explanations on model performance

[10]:
```python
# <Student to fill this section>
for n_neighbors, p in product(n_neighbors_list, p_list):
    mse = mean_squared_error(y_val, predictions[f"y_pred_{n_neighbors}_{p}"])
    rmse = np.sqrt(mse)
    mae = mean_absolute_error(y_val, predictions[f"y_pred_{n_neighbors}_{p}"])
    r2 = r2_score(y_val, predictions[f"y_pred_{n_neighbors}_{p}"])

    print(f"n_neighbors = {n_neighbors}, p = {p}")
    print(f"RMSE: {rmse}")
    print(f"MAE: {mae}")
    print(f"R2: {r2}")
    print("............................")
```

```
n_neighbors = 1, p = 1
RMSE: 10.940706240014995
MAE: 5.602651515151515
R2: 0.7685414511302396
```

```
…
n_neighbors = 1, p = 2
RMSE: 11.219428157522922
MAE: 5.686742424242424
R2: 0.7565981048977521
…
n_neighbors = 2, p = 1
RMSE: 10.565826757795984
MAE: 7.041477272727272
R2: 0.7841313962917543
…
n_neighbors = 2, p = 2
RMSE: 10.6774374479341
MAE: 7.043939393939394
R2: 0.7795467107266877
…
n_neighbors = 3, p = 1
RMSE: 10.641148848763878
MAE: 7.302777777777776
R2: 0.7810426401932502
…
n_neighbors = 3, p = 2
RMSE: 10.800668508740104
MAE: 7.376388888888888
R2: 0.7744287296129941
…
n_neighbors = 5, p = 1
RMSE: 10.802706941907072
MAE: 7.549924242424243
R2: 0.7743435764857882
…
n_neighbors = 5, p = 2
RMSE: 10.824338456935973
MAE: 7.54878787878788
R2: 0.7734389555365734
…
n_neighbors = 7, p = 1
RMSE: 10.958514774958084
MAE: 7.683008658008658
R2: 0.7677873330195206
…
n_neighbors = 7, p = 2
RMSE: 11.035772250789922
MAE: 7.733170995670995
R2: 0.7645015951411606
…
n_neighbors = 9, p = 1
RMSE: 11.096640513950488
```

```
MAE: 7.692382154882155
R2: 0.7618966284044159
…
n_neighbors = 9, p = 2
RMSE: 11.28316576937848
MAE: 7.875
R2: 0.7538247153040788
…
n_neighbors = 11, p = 1
RMSE: 11.20004205174836
MAE: 7.766769972451789
R2: 0.7574385288763921
…
n_neighbors = 11, p = 2
RMSE: 11.436955343673555
MAE: 7.936088154269974
R2: 0.7470682422625163
…
```

[19]:

```python
# @title Model Performance Explanation

wgt_model_performance_explanation = widgets.Textarea(
    value="The best result was achieved with n_neighbors = 2 and p = 1,␣
 ↪yielding an RMSE of 10.5658. Using n_neighbors = 2 allows the model to␣
 ↪capture local patterns while reducing noise compared to n_neighbors = 1,␣
 ↪which may lead to overfitting. The choice of p = 1 (Manhattan Distance)␣
 ↪suggests that absolute differences between feature values are more effective␣
 ↪for this dataset.",
    placeholder='<student to fill this section>',
    description='Model Performance Explanation:',
    disabled=False,
    style={'description_width': 'initial'},
    layout=widgets.Layout(height="100%", width="auto")
)
wgt_model_performance_explanation
```

```
Textarea(value='The best result was achieved with n_neighbors = 2 and p = 1,␣
 ↪yielding an RMSE of 10.5658. Usin…
```

[20]:

```python
print('Model Performance Explanation:', wgt_model_performance_explanation.value)
```

```
Model Performance Explanation: The best result was achieved with n_neighbors = 2
and p = 1, yielding an RMSE of 10.5658. Using n_neighbors = 2 allows the model
to capture local patterns while reducing noise compared to n_neighbors = 1,
which may lead to overfitting. The choice of p = 1 (Manhattan Distance) suggests
that absolute differences between feature values are more effective for this
dataset.
```

1.6.5 E.5 Business Impact from Current Model Performance

Provide some analysis on the model impacts from the business point of view

[ ]:
```
# <Student to fill this section>

y_1 = y_train['rent']
y_pred = predictions[f"y_pred_{2}_{1}"]
y_pred = pd.DataFrame(y_pred, columns=['rent_pred'])
y_2 = y_pred['rent_pred']

train_set = alt.Chart(pd.DataFrame({'target': y_1, 'preds': y_1})).
 ↪mark_line(color='green').encode(
    x='target',
    y='preds'
)
test_set = alt.Chart(pd.DataFrame({'target': y_1, 'preds': y_2})).mark_line().
 ↪encode(
    x='target',
    y='preds'
  )

test_set + train_set
```

[ ]: alt.LayerChart(…)

[11]:
```
model_final = KNeighborsRegressor(n_neighbors=2, p=1)
model_final.fit(X_train, y_train)

y_pred_final = model_final.predict(X_test)

mse_final = mean_squared_error(y_test, y_pred_final)
rmse_final = np.sqrt(mse_final)
mae_final = mean_absolute_error(y_test, y_pred_final)
r2_final = r2_score(y_test, y_pred_final)

print(f"RMSE: {rmse_final}")
print(f"MAE: {mae_final}")
print(f"R2: {r2_final}")
```

```
RMSE: 22.7831456030172
MAE: 13.91758443465492
R2: 0.6065824377689983
```

[21]:
```
# @title Model Business Impacts Explanation

wgt_model_business_explanation = widgets.Textarea(
```

```
    value="The current K-Nearest Neighbors (KNN) model with n_neighbors=2 and␣
↪p=1 achieved an RMSE of 22.78 and an R² of 0.61 on unseen data, indicating␣
↪moderate error and explaining 61% of the variance. However, the model may be␣
↪overfitted, as it performed better on the training set. The increase in RMSE␣
↪and decrease in R² on the test set suggests a drop in predictive accuracy,␣
↪which is common in real-world applications. Despite this, the model provides␣
↪a solid foundation for predicting rental prices, though further refinement␣
↪is needed for more accurate predictions.",
    placeholder='<student to fill this section>',
    description='Model Business Impacts Explanation:',
    disabled=False,
    style={'description_width': 'initial'},
    layout=widgets.Layout(height="100%", width="auto")
)
wgt_model_business_explanation
```

```
Textarea(value='The current K-Nearest Neighbors (KNN) model with n_neighbors=2␣
↪and p=1 achieved an RMSE of 22.…
```

[22]:
```
print('Model Business Impacts Explanation:', wgt_model_business_explanation.
↪value)
```

```
Model Business Impacts Explanation: The current K-Nearest Neighbors (KNN) model
with n_neighbors=2 and p=1 achieved an RMSE of 22.78 and an R² of 0.61 on unseen
data, indicating moderate error and explaining 61% of the variance. However, the
model may be overfitted, as it performed better on the training set. The
increase in RMSE and decrease in R² on the test set suggests a drop in
predictive accuracy, which is common in real-world applications. Despite this,
the model provides a solid foundation for predicting rental prices, though
further refinement is needed for more accurate predictions.
```

1.7 F. Experiment Outcomes

[ ]:
```
# @title Experiment Outcomes Explanation

wgt_experiment_outcomes_explanation = widgets.Select(
    options=['Hypothesis Confirmed', 'Hypothesis Partially Confirmed',␣
↪'Hypothesis Rejected'],
    value='Hypothesis Rejected',
    description='Experiment Outcomes:',
    disabled=False,
)

wgt_experiment_outcomes_explanation
```

```
Select(description='Experiment Outcomes:', index=2, options=('Hypothesis␣
↪Confirmed', 'Hypothesis Partially Con…
```

[24]:
```
# @title Experiments Results Explanation

wgt_experiment_results_explanation = widgets.Textarea(
    value="Hypothesis Partially Confirmed. The experiment with different␣
  ↪combinations of n_neighbors and p showed that smaller values of n_neighbors␣
  ↪(such as 2 and 3) resulted in lower RMSE, indicating better performance.␣
  ↪Specifically, n_neighbors = 2 and p = 1 produced the lowest RMSE of 10.5658,␣
  ↪suggesting this as the optimal configuration. This indicates that a small␣
  ↪number of neighbors captures local patterns effectively, and Manhattan␣
  ↪Distance (p = 1) outperforms Euclidean Distance (p = 2). As n_neighbors␣
  ↪increased, RMSE rose, indicating a shift toward underfitting. Overall, the␣
  ↪current KNN model performs well for rental price prediction, but there is␣
  ↪room for improvement. The drop in performance from the validation to the␣
  ↪test set suggests that further tuning or exploring alternative models could␣
  ↪be beneficial. From a business perspective, even with the modest error␣
  ↪margin, the model's predictions are valuable for strategic decision-making␣
  ↪and can help optimise rental pricing strategies. Future steps could involve␣
  ↪exploring new models and features to improve prediction accuracy.",
    placeholder='<student to fill this section>',
    description='Experiments Results Explanation:',
    disabled=False,
    style={'description_width': 'initial'},
    layout=widgets.Layout(height="100%", width="auto")
)
wgt_experiment_results_explanation
```

```
Textarea(value='Hypothesis Partially Confirmed. The experiment with different␣
 ↪combinations of n_neighbors and …
```

[25]:
```
print('Experiments Results Explanation:', wgt_experiment_results_explanation.
 ↪value)
```

```
Experiments Results Explanation: Hypothesis Partially Confirmed. The experiment
with different combinations of n_neighbors and p showed that smaller values of
n_neighbors (such as 2 and 3) resulted in lower RMSE, indicating better
performance. Specifically, n_neighbors = 2 and p = 1 produced the lowest RMSE of
10.5658, suggesting this as the optimal configuration. This indicates that a
small number of neighbors captures local patterns effectively, and Manhattan
Distance (p = 1) outperforms Euclidean Distance (p = 2). As n_neighbors
increased, RMSE rose, indicating a shift toward underfitting. Overall, the
current KNN model performs well for rental price prediction, but there is room
for improvement. The drop in performance from the validation to the test set
suggests that further tuning or exploring alternative models could be
beneficial. From a business perspective, even with the modest error margin, the
model's predictions are valuable for strategic decision-making and can help
optimise rental pricing strategies. Future steps could involve exploring new
models and features to improve prediction accuracy.
```