Homework 2

Casey Bramlett

CS488 Intro to Big Data

2/24/25

1a. Heatmaps and pairplot

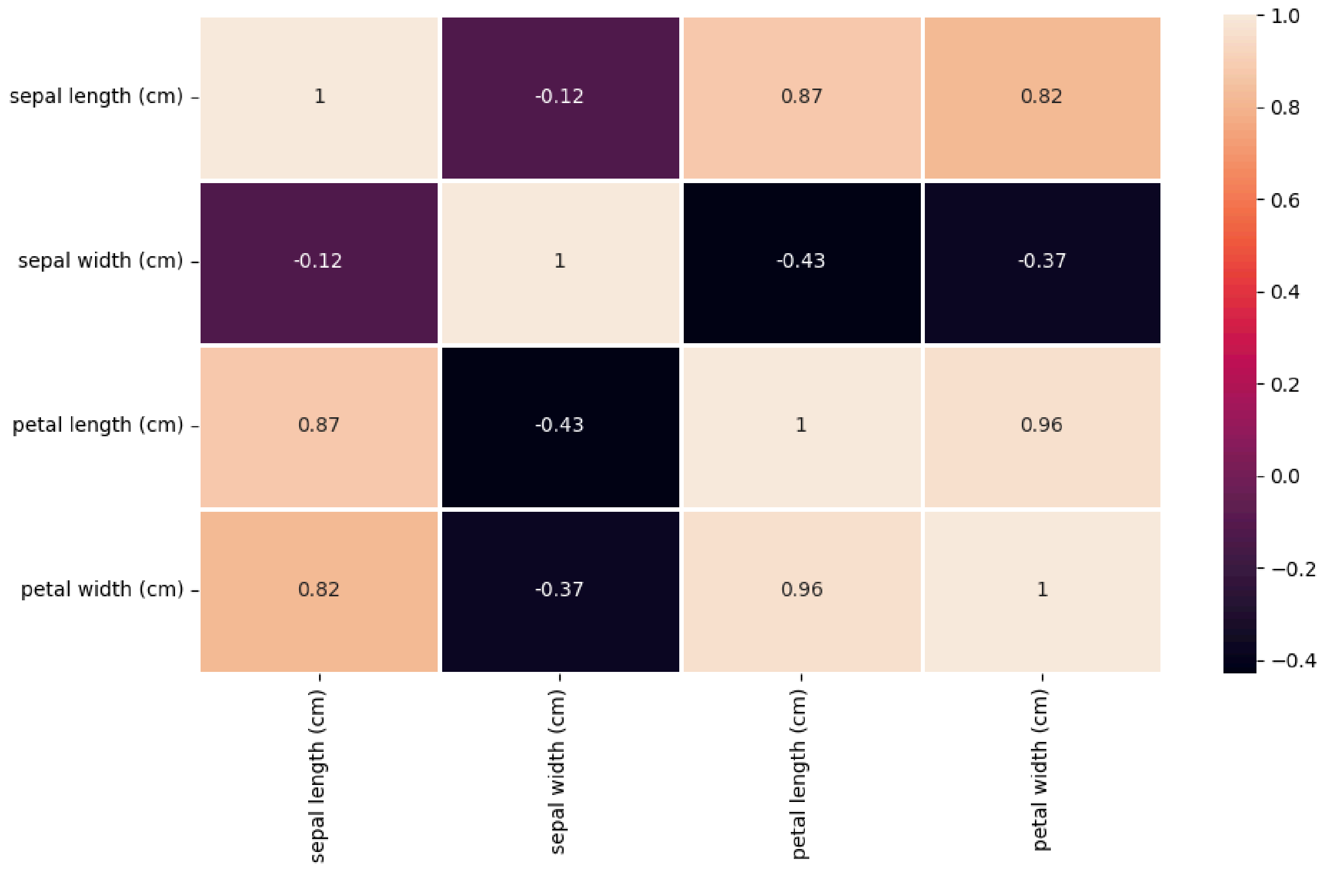

Figure 1.1 - Correlation Coefficient Heatmap

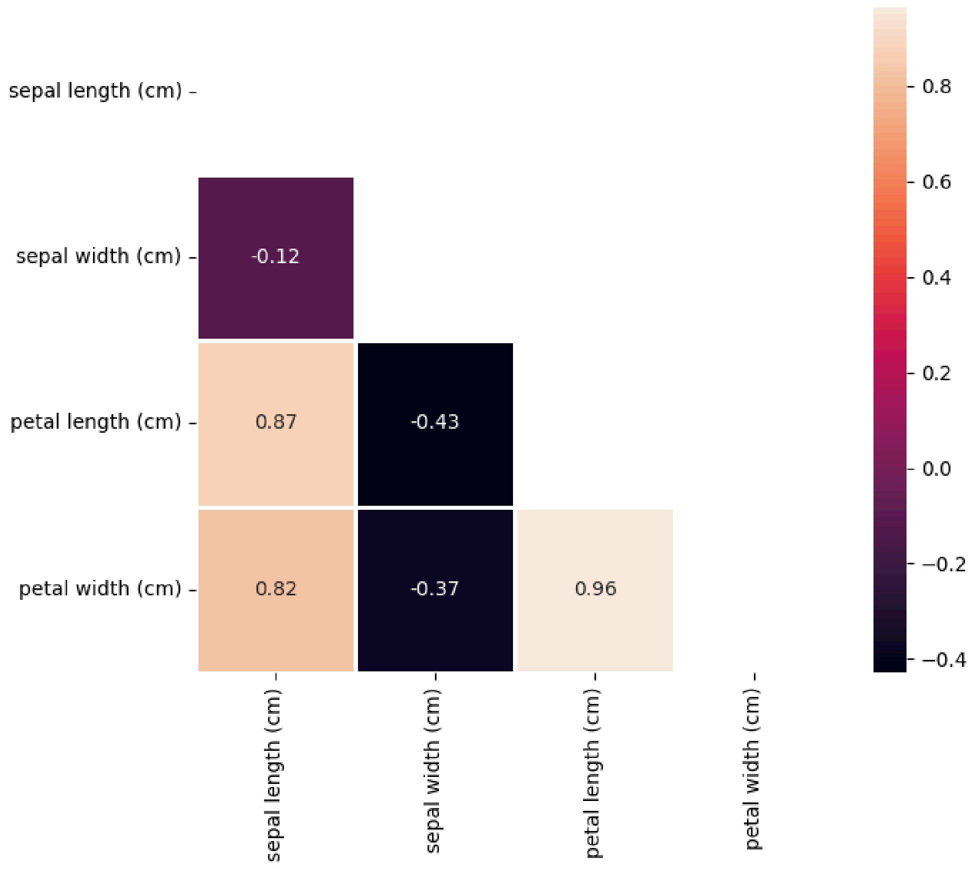

Figure 1.2 - Lower Half of Correlation Coefficient Heatmap

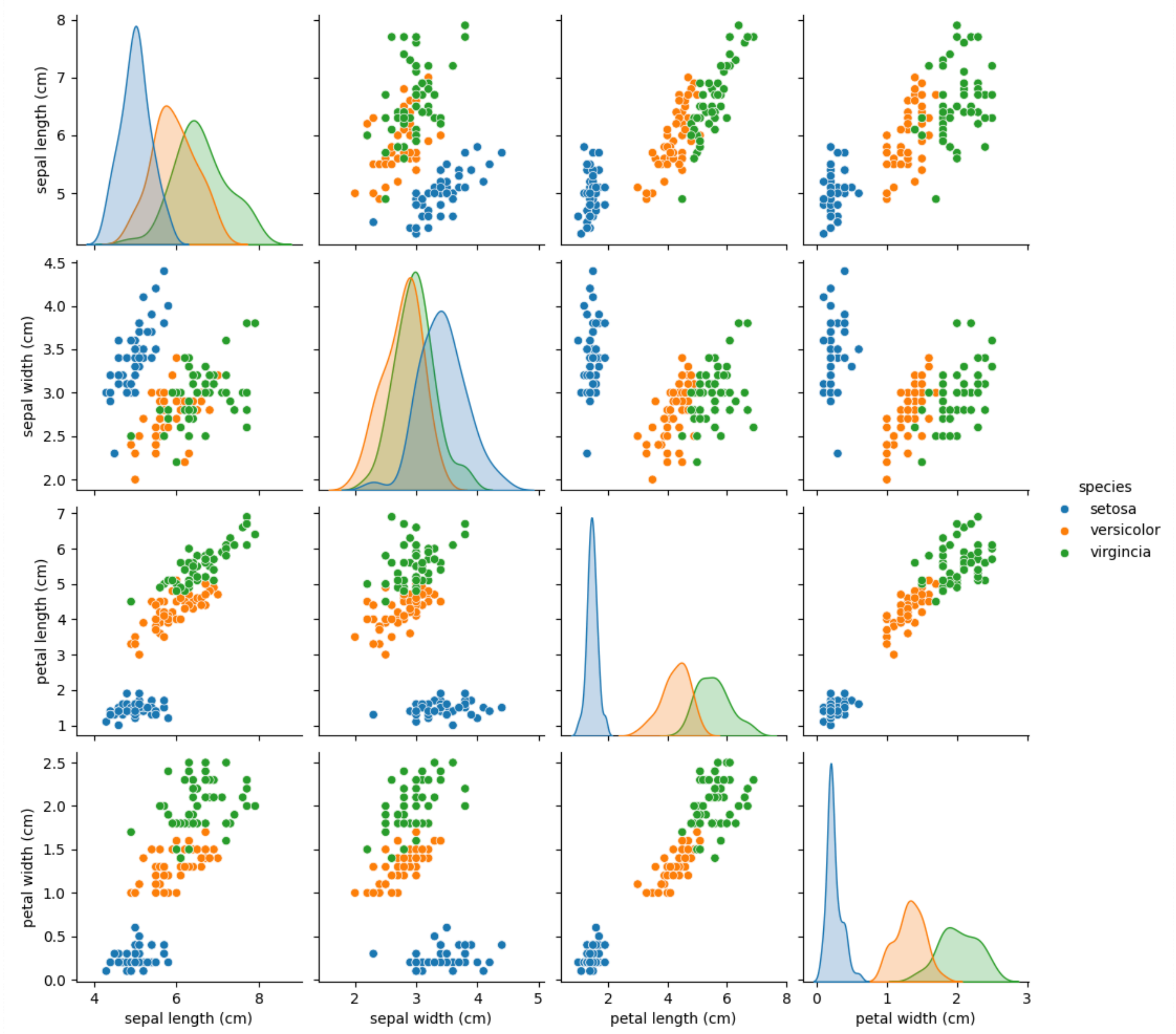

Figure 1.3 Pairplot

1b. Data analysis

a) Implications of data distribution on data analysis

From the histograms and pairplots we can tell that petal length and petal width have very distinct distributions for each species. Particularly 'setosa' usually has smaller petals compared to 'versicolor' and 'virginica'.

Sepal width shows more overlap among the 3 species, meaning it could make less discriminative feature in classification. These distributions suggest that 'setosa' is more easily separable from the other two species based on petal distributions, where as the other two have more overlap.

In terms of implications:

Model Selection and Complexity:

Highly correlated features (like petal length and petal width) may introduce collinearity issues in linear models.Dimensionality reduction or feature selection might be needed or a good idea if the model needs to be simpler.

Data Preprocessing:

The distinct distributions could allow simpler models to classify 'setosa' well, but distinguishing 'versicolor' from 'virginica' may require more complicated apporeaches and or more a more detailed look at other features.

b) Inferences from (i) Correlation Heat Map and (ii) Feature Plots

Correlation Heat Map:

Petal length and petal width show a very high positive correlation (0.96). This implies these two features largely change together. Sepal length also correlates strongly with both petal length (0.87) and petal width (0.82). This suggests a that there could be redundancy if these three features were used in certain predictive models.

Sepal width is negatively related to the other features, although the correlation is weaker.

Influence on Analysis:

Because of the strong correlation among petal dimensions (and with sepal length), you might not need all three features (petal length, petal width, sepal length) if your goal is a simpler model.

However, if interpretability or small performance gains matter in your specific scenario, you could still keep them.

Feature Plots (Pairplot):

The pairplot reveals that 'setosa' points are clustered distinctly, especially along petal length and petal width. 'Versicolor' and 'virginica' overlap more in sepal features but separate more clearly on the petal features.

The distributions along the diagonal in the pairplot also confirm that petal measurements provide better separations than sepal measurements for distinguishing species.

Influence on Analysis:

Models aiming to separate 'setosa' from the other classes can rely heavily on petal dimensions. Distinguishing 'versicolor' and 'virginica' may require more complex use of multiple features.

Overall, the data visualizations suggest that petal features are very helpful for species identification, while sepal width is less correlated with the other dimensions and may provide additional, although weaker, discriminative power. These insights can help guide model design and feature selection when cleaning data and figuring out other predictive tasks.

2. Use Iris dataset for a Linear Regression (LR) analysis

a) LR Parameter Output

```
Case i) 20% Training Samples:
Intercept: -0.22951225701028832
Coefficients: [ 0.71861432 -0.66987086  1.55485573]
RMSE: 0.3280642841209079

For sample index 73:
Actual petal length (cm): 4.7
Predicted petal length (cm): 4.144223569395406


Case ii) 80% Training Samples
Intercept: -0.26219590258870795
Coefficients: [ 0.72281463 -0.63581649  1.46752403]
RMSE: 0.360577675839516

For sample index 73:
Actual petal length (cm): 4.7
Predicted petal length (cm): 4.127715970146925
```

b) Analysis On Results

Similarities:

Both models have similar intercepts and coefficients. This indicates that the relationship between the predictors (sepal length, sepal width, petal width) and the target (petal length) is consistent regardless of whether you use 20% or 80% of the data for training.

RMSE Comparison:

The RMSE values are very close, about 0.33 for 20% training and 0.36 for 80% training. In this certain split, the model trained on 20% of the data produced a somewhat lower RMSE, meaning its predictions were a little closer on average. However, these differences are small and could be due to random variations in the split.

Prediction on Sample Index 73:

In both cases, for the chosen sample (index 73), the predicted petal lengths are slightly lower than the actual value of 4.7 cm. The predictions are 4.144 cm (20% case) and 4.128 cm (80% case), which are very close.

Both models learn almost the same relationship between the features and petal length. Small differences in RMSE (0.33 vs. 0.36) show that the model with 20% training data performed a tiny bit better on average, but the difference is so small that both models are very similar.

For the specific sample (index 73), both models underestimated the petal length by about 0.55 to 0.57 cm.

Overall, these results illustrate that even with different training sizes, the learned relationships and prediction errors are very similar. The small differences might come from the specific random splits used for training and testing. With these results there isn't a clear better or worse case since they are so similar. However, I would say the 20% training data case would be more efficient since it would basically get the same results with less data needed.

Appendix A

Homework2.py (for 1a-1b)

```python
from sklearn.datasets import load_iris
import numpy as np
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt

# load iris data
iris = load_iris()

#create pd dataframes
iris_df = pd.DataFrame(data=iris.data, columns = iris.feature_names)
target_df = pd.DataFrame(data= iris.target, columns= ['species'])

# generate labels
def converter(specie):
    if specie ==0:
        return 'setosa'
    elif specie == 1:
        return 'versicolor'
    else:
        return 'virgincia'
target_df['species'] = target_df['species'].apply(converter)
```

```python
#concat the dataframes
df = pd.concat([iris_df, target_df], axis=1)

#display data headers
#df.head

df.info()

#display random data samples
df.sample(10)

#display data columns
df.columns

#disaply the size of the dataset
df.shape

#output data

print(df)

#display correlation coeff
# df.corr()

#visualize iris features as a heatmap
# cor_eff=df.corr()
cor_eff = df.select_dtypes(include=[np.number]).corr()
print(cor_eff)


# visualize the correlation heatmap
plt.figure(figsize=(6,6))
sns.heatmap(cor_eff, linecolor='white', linewidths=1, annot=True)
# plt.show()  # display the heatmap

# plot the lower half of the correlation matrix
fig, ax = plt.subplots(figsize=(6,6))
mask = np.zeros_like(cor_eff)
mask[np.triu_indices_from(mask)] = 1
sns.heatmap(cor_eff, linecolor='white', linewidths=1, mask=mask, ax=ax,
annot=True)
# plt.show()  # display the lower half heatmap

# create the pairplot
g = sns.pairplot(df, hue='species')
```

```
# plt.show()  # display the pairplot
plt.show() #show all plots at once
```

Homework2_Iris_LR.py for part 2 (case i and case ii and a & b)

```python
from sklearn.datasets import load_iris
from sklearn.linear_model import LinearRegression
from sklearn.model_selection import train_test_split
from sklearn.metrics import mean_squared_error
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt


# load iris and create dataframe
iris = load_iris()
# 4 columns
iris_df = pd.DataFrame(data=iris.data, columns=iris.feature_names)

# prep for linear regression analysis
# X: predictors (all features except petal length)
X = iris_df[['sepal length (cm)', 'sepal width (cm)', 'petal width (cm)']]
# y: target variable - petal length
y = iris_df['petal length (cm)']

# case i) train the model on 20% of the samples
X_train_20, X_test_20, y_train_20, y_test_20 = train_test_split(
    X, y, train_size=0.2, random_state=42
)

# 20% training data
lr_20 = LinearRegression()
lr_20.fit(X_train_20, y_train_20)

# predict petal length on the test set
y_pred_20 = lr_20.predict(X_test_20)
# calculate RMSE for 20% training
rmse_20 = np.sqrt(mean_squared_error(y_test_20, y_pred_20))

# LR parameters and RMSE for 20% training
print("Case i) 20% Training Samples:")
print("Intercept:", lr_20.intercept_)
print("Coefficients:", lr_20.coef_)
print("RMSE:", rmse_20)
```

```
# choose a sample from the test set that was not in training
sample_index_20 = X_test_20.index[0]
# reshape
sample_20 = X_test_20.loc[[sample_index_20]] #double blackets to keep as
dataframe
predicted_value_20 = lr_20.predict(sample_20)[0]

# sample_20 = X_test_20.loc[sample_index_20].values.reshape(1, -1)
# predicted_value_20 = lr_20.predict(sample_20)[0]
actual_value_20 = y_test_20.loc[sample_index_20]

print("\nFor sample index {}:".format(sample_index_20))
print("Actual petal length (cm):", actual_value_20)
print("Predicted petal length (cm):", predicted_value_20)
print("\n")

# case ii) train the model on 80% of the samples
X_train_80, X_test_80, y_train_80, y_test_80 = train_test_split(
    X, y, train_size=0.8, random_state=42
)

# 80% training data
lr_80 = LinearRegression()
lr_80.fit(X_train_80, y_train_80)

# predict petal length on the test set
y_pred_80 = lr_80.predict(X_test_80)
# calculate RMSE for 80% training
rmse_80 = np.sqrt(mean_squared_error(y_test_80, y_pred_80))

#  LR parameters and RMSE for 80% training
print("Case ii) 80% Training Samples")
print("Intercept:", lr_80.intercept_)
print("Coefficients:", lr_80.coef_)
print("RMSE:", rmse_80)

# choose a sample from the test set that was not in training
sample_index_80 = X_test_80.index[0]
sample_80 = X_test_80.loc[[sample_index_80]]  #double blackets to keep as
dataframe
predicted_value_80 = lr_80.predict(sample_80)[0]

# sample_80 = X_test_80.loc[sample_index_80].values.reshape(1, -1)
# predicted_value_80 = lr_80.predict(sample_80)[0]
actual_value_80 = y_test_80.loc[sample_index_80]
```

```
print("\nFor sample index {}:".format(sample_index_80))
print("Actual petal length (cm):", actual_value_80)
print("Predicted petal length (cm):", predicted_value_80)
```

Appendix B (data dumps and console outputs)

Iris DataFrame Sample

Index	sepal length (cm)	sepal width (cm)	petal length (cm)	petal width (cm)	species
0	5.1	3.5	1.4	0.2	setosa
1	4.9	3.0	1.4	0.2	setosa
2	4.7	3.2	1.3	0.2	setosa
3	4.6	3.1	1.5	0.2	setosa
4	5.0	3.6	1.4	0.2	setosa

Correlation Matrix

	sepal length (cm)	sepal width (cm)	petal length (cm)	petal width (cm)
sepal length (cm)	1.000000	-0.117570	0.871754	0.817941
sepal width (cm)	-0.117570	1.000000	-0.428440	-0.366126
petal length (cm)	0.871754	-0.428440	1.000000	0.962865
petal width (cm)	0.817941	-0.366126	0.962865	1.000000