DATASCI207-005/007 Applied Machine Learning

Vilena Livinsky, PhD(c)

School of Information, UC Berkeley

Week 3: 01/22/2025 & 01/23/2025

Today's Agenda

- Review: Linear Regression & Gradient Descent
- Feature Engineering
- Walkthroughs:
  - Feature Engineering
  - TensorFlow

Final Project: Step 1 (Reminder)

Form a group

- 3-4 people, max 4
- NO silos or groups of 2
- Groups can only be composed of you and your colleagues in your section

Inform me & the class of your formed group in Slack

- Include names of group members
- Due date: 01/24/2025 EOD

General Plan:

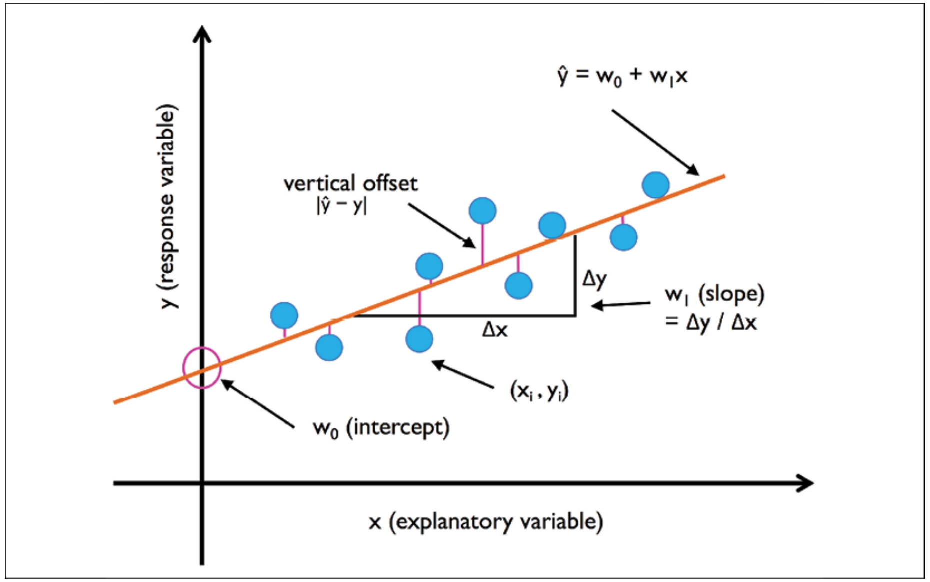
- Step 1: form a group
- Step 2: submit your group’s question to investigate + dataset
- Step 3: a baseline presentation
- Step 4: final presentation

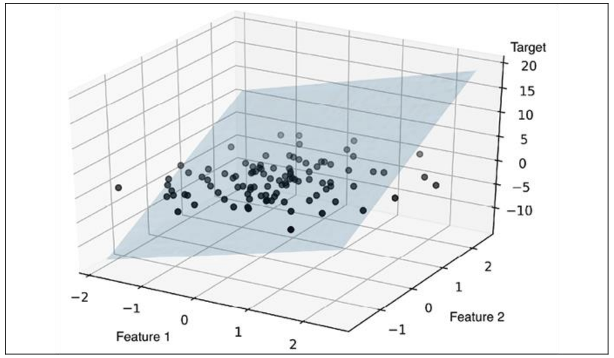
Linear Regression & Gradient Descent (Review)

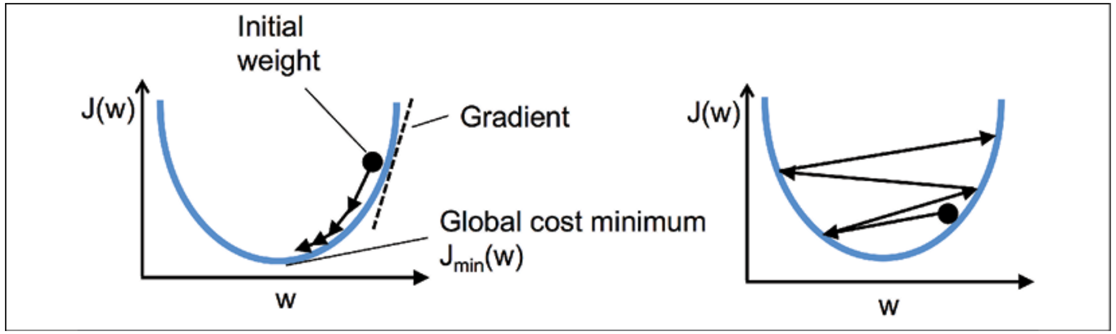
- Training:
  - Get data
  - Initialize weight/s
  - Initialize bias
- Given data
  - Get prediction
  - Get error
  - Update weight, use Gradient Descent
  - Repeat

```python
# Run gradient descent
learning_rate = 0.1

preds = np.dot(X, W)
loss = ((preds - Y)**2).mean()
gradient = 2 * np.dot((preds - Y), X) / m
W = W - learning_rate * gradient

print('predictions:', preds)
print('loss:', loss)
print('gradient:', gradient)
print('weights:', W)
```

```
predictions: [6 5]
loss: 16.0
gradient: [ 8. 20. 12.  4.]
weights: [ 0.2 -1.  -0.2  0.6]
```

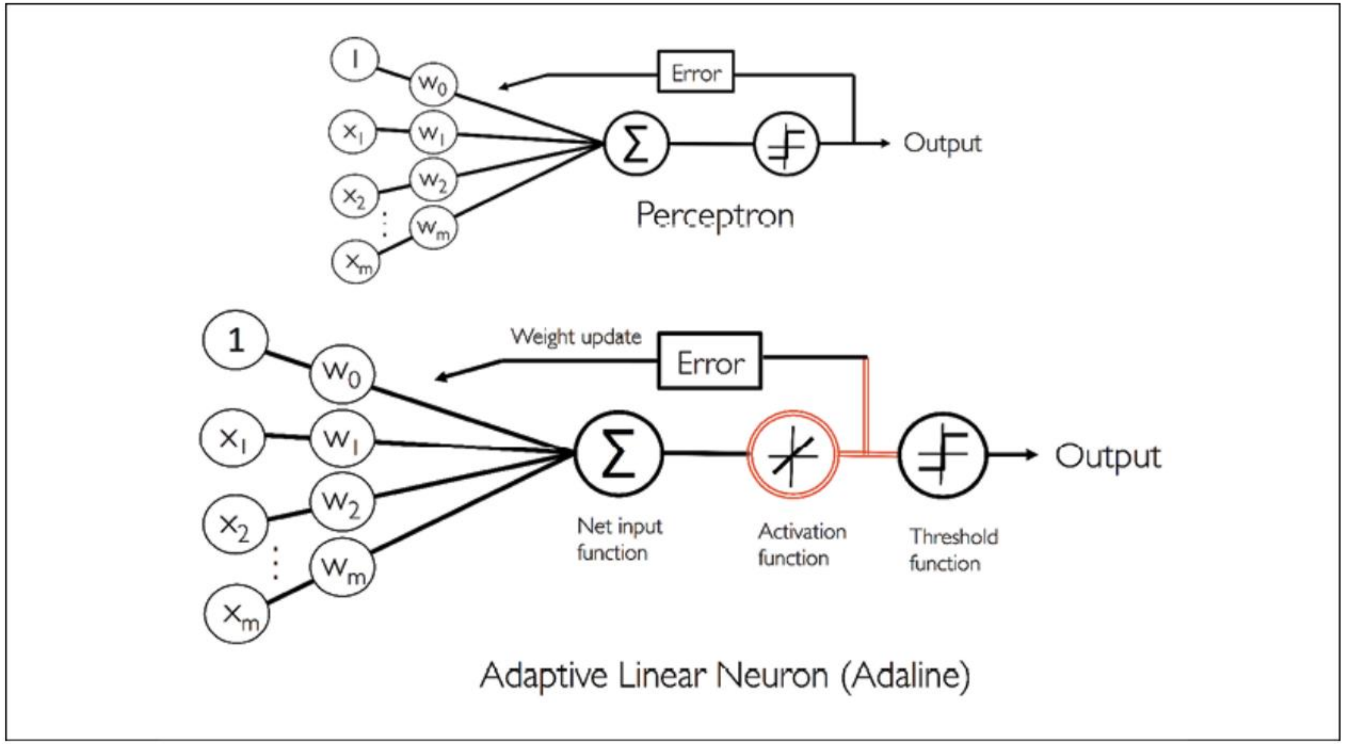






Raschka, S., & Mirjalili, V. (2019). Python Machine Learning, Third Edit.

Single Layer Neural Net & Activation Function(s)



Raschka, S., & Mirjalili, V. (2019). Python Machine Learning, Third Edit.

net input $z = \boldsymbol{w}^T\boldsymbol{x}$

↓

linear activation function, $\phi(z)$,

$$\phi(\boldsymbol{w}^T\boldsymbol{x}) = \boldsymbol{w}^T\boldsymbol{x}$$

Practice

What is the activation function for Linear Regression? What does it do?

Consider:

net input $z = \boldsymbol{w}^T \boldsymbol{x}$

linear activation function, $\phi(z)$,

$$\phi(\boldsymbol{w}^T \boldsymbol{x}) = \boldsymbol{w}^T \boldsymbol{x}$$

Look up in Keras API docs:
-look in “layer activation functions” section
- what is a “linear” activation function? what does it do?

-look at the “Dense layer” section
- What does *activation=None* mean?

```
def build_model(num_features):
    """Return a simple linear regression model using the Keras Sequential API."""
  # Clear session and set a random seed for consistent behavior.
    tf.keras.backend.clear_session()
    tf.random.set_seed(0)

  # Use Keras Sequential API to build a linear regression model.
    model = keras.Sequential()

    # create input layer
    model.add(tf.keras.Input(shape=(num_features,),
         name='Input'
    ))


    # create output layer
    model.add(keras.layers.Dense(
      activation = None,           # __?__ activation is used
      units=1,                     # there is a single output
      use_bias=True                # include a learned bias parameter
      ))



  # Use mean squared error as our loss and the Adam optimizer.
    optimizer = tf.keras.optimizers.Adam(learning_rate=0.01)
    model.compile(loss='mse', optimizer=optimizer)

    return model

# Build a model
model = build_model(num_features=temp_X_train.shape[1])
```

Gradient Descent

- Consider (ex. Linear Regression):
  - What is the advantage of having a continuous linear activation function? (Hint: consider this in the context of the cost function)
  - What about the cost function being convex?

Regression Evaluation Metrics

Squared Loss

$$J = \sum_{n=1}^{N} \left(y^{(n)} - f(\mathbf{x}^{(n)}; \mathbf{w}) \right)^2$$

Mean Squared Error (MSE)

Mean Absolute Error (MAE)

Regression Evaluation Metrics

Squared Loss

$$J = \sum_{n=1}^{N} \left(y^{(n)} - f(\mathbf{x}^{(n)}; \mathbf{w}) \right)^2$$

Mean Squared Error (MSE)

$$MSE = \frac{1}{N}\Sigma_{i=1}^{N}(y_i - \hat{y}_i)^2$$

$$J(w_0, w_1) = \frac{1}{m}\sum_i (w_0 + w_1 x_i - y_i)^2$$

- Accounts for data size
- Penalizes large outliers (sensitive to outliers)

Root Mean Squared Error (RMSE)

$$RMSE = \sqrt{\frac{1}{N}\Sigma_{i=1}^{N}(y_i - \hat{y}_i)^2}$$

- Scaled to outcome variable

Mean Absolute Error (MAE)

$$MAE = \frac{1}{N}\Sigma_{i=1}^{N}|y_i - \hat{y}_i|$$

- Robust to outliers
- Uses absolute value such that large errors do not disproportionately affect the overall loss (vs. MSE where we have squared errors)

Consider:
- Sensitivity of the loss function to outliers
  - Do we want to penalize outliers/skewed data points during training?
- Interpretability
- Some loss functions are more computationally intensive
- Convergence properties: Convexity of a loss function

Visualizing our data

Where to start?

Visualise your data!

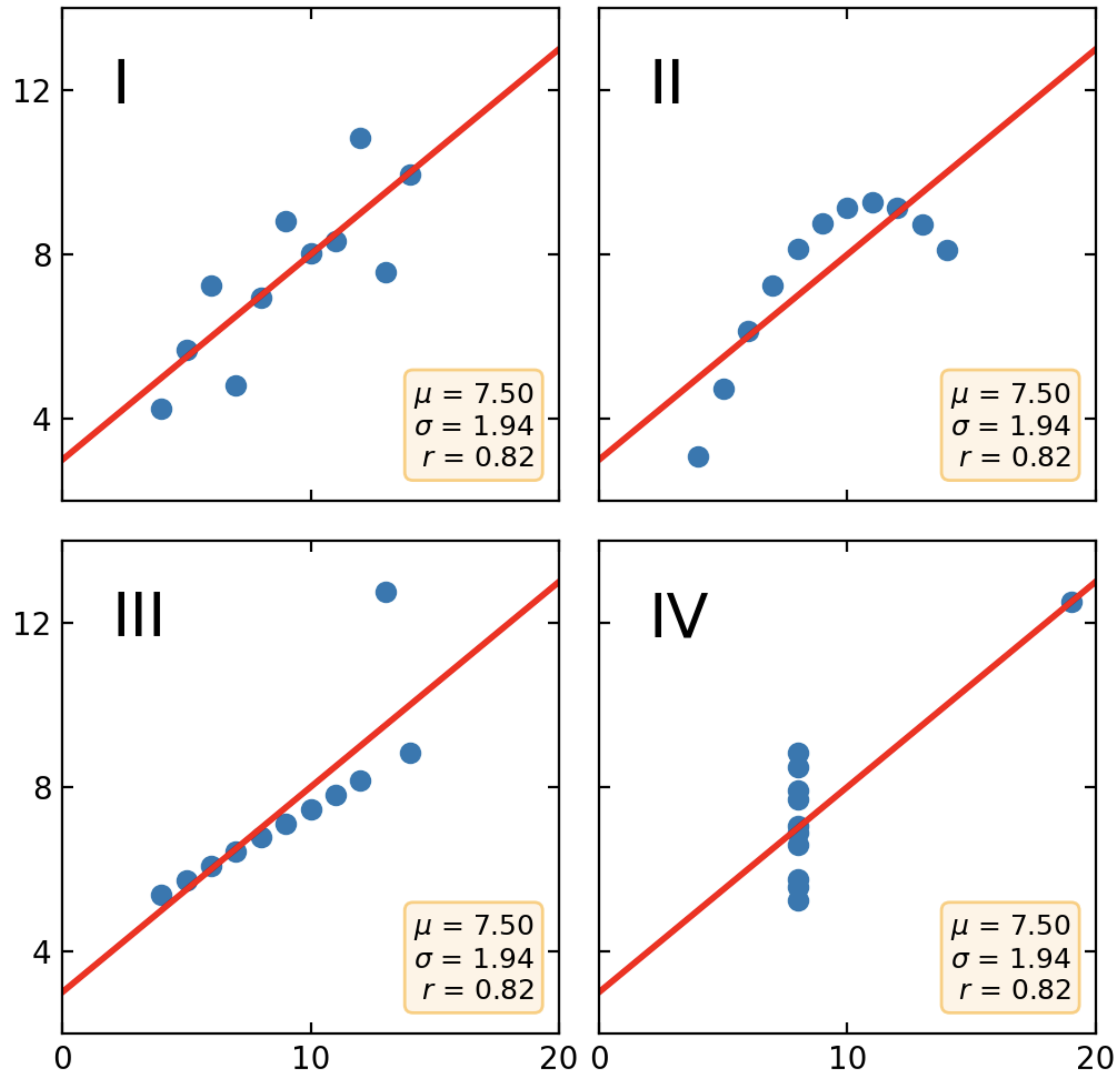
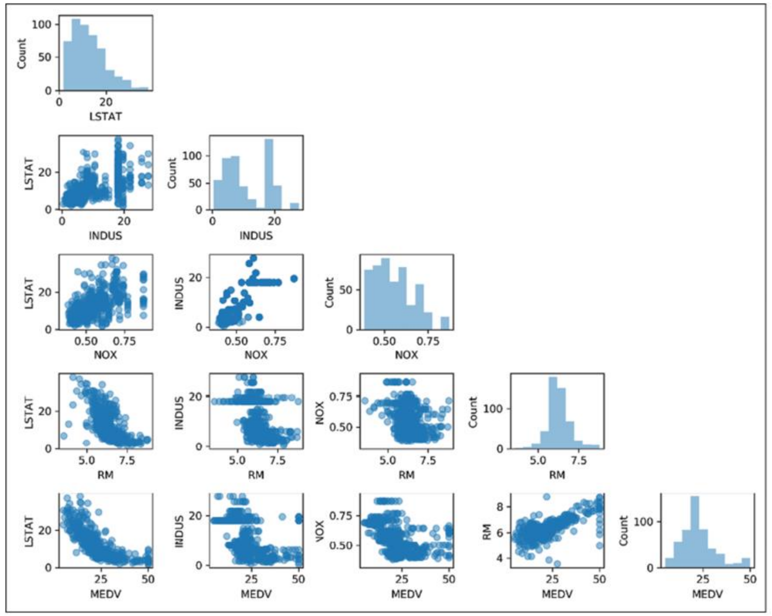


Image Source: https://matplotlib.org/stable/gallery/specialty_plots/anscombe.html

Data Visualization

Scatterplot matrix

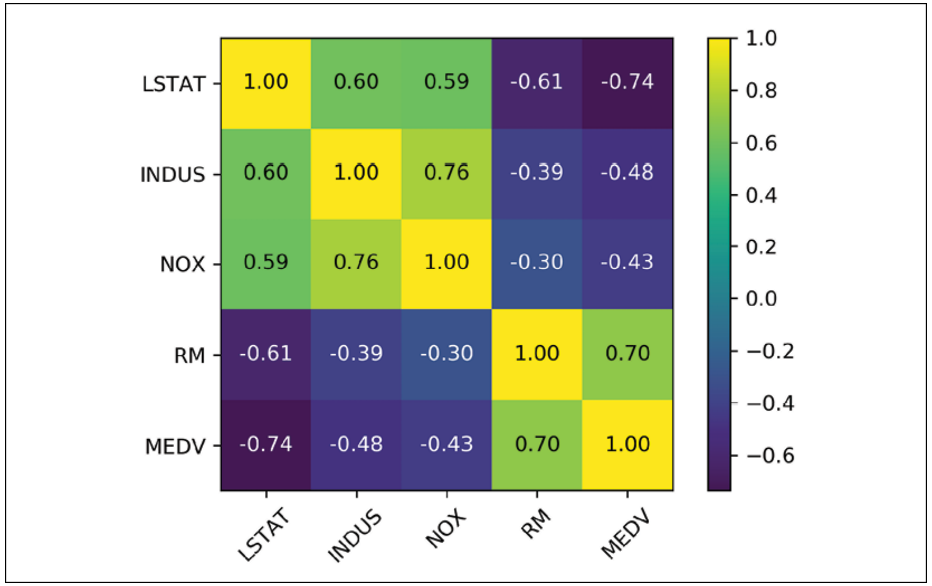
- graphical summary of the relationships



- Chapter Exs.:
  - https://learning.oreilly.com/library/view/python-machine-learning/9781789955750/Text/Chapter_10.xhtml#_idParaDest-194
- Corresponding Git:
  - https://github.com/rasbt/python-machine-learning-book-3rd-edition/blob/master/ch10/ch10.ipynb

Correlation Matrix

- The correlation matrix
  - square matrix
  - Pearson product-moment correlation coefficient (Pearson's r)
    - measures the linear dependence between pairs of features
    - range –1 to 1
- correlation matrix array as a heat map:



Feature Engineering

Feature Engineering: Example 1

ID	Country	Age	Income	Loan_Approved
1	United States	25	50000	Yes
2	United States	30	60000	No
3	United States	22	40000	Yes
4	United States	28	45000	Yes
5	United States	35	70000	No
6	United States	40	75000	Yes
7	United States	29	52000	No
8	United States	33	68000	Yes
9	United States	26	47000	No
10	United States	31	65000	Yes

Which fields (columns) might be on first sight usable for building a predictive model?

- Target variable: *Loan_Approved*

Feature Engineering: Example 2

Raw Data Field	Derived Feature Name	Description	Missing Val Reason	Imputation Strategy

Goal:	Anomaly Detection
Data for the model build:	historical records
Time frame for training:	?
Time frame for testing:	?

```
root
 |-- time: string (nullable = true)
 |-- user_id: integer (nullable = true)
 |-- attachment: string (nullable = true)
```

```
+---------------------+-------+--------------------------------+
| time                |user_id| attachment                     |
+---------------------+-------+--------------------------------+
| 2025-01-23 16:30:00 |  1001 | quarterly_report.pdf           |
| 2025-01-23 16:32:00 |  1002 | budget_projection_2025.xlsx    |
| 2025-01-23 16:34:00 |  1003 | client_presentation.pptx       |
| 2025-01-23 16:36:00 |  1004 | employee_handbook_2025.pdf     |
| 2025-01-23 16:38:00 |  1005 | investment_analysis.docx       |
| 2025-01-23 16:40:00 |  1006 | financial_report_2024.pdf      |
| 2025-01-23 16:42:00 |  1007 | project_plan_Q1_2025.xlsx      |
| 2025-01-23 16:44:00 |  1008 | HR_policies_2025.pdf           |
| 2025-01-23 16:46:00 |  1009 | IT_security_policies.pdf       |
| 2025-01-23 16:48:00 |  1010 | compliance_audit_report.pdf    |
+---------------------+-------+--------------------------------+
```

Feature Engineering: Example 2

Raw Data Field	Derived Feature Name	Description	Missing Val Reason	Imputation Strategy
user_id, time	login_count	Num of logins for a user on day		

Goal: Anomaly Detection

Data for the model build: historical records

Time frame for training: ?

Time frame for testing: ?

```
root
 |-- time: string (nullable = true)
 |-- user_id: integer (nullable = true)
 |-- attachment: string (nullable = true)
```

```
+-------------------+-------+------------------------------+
| time              |user_id| attachment                   |
+-------------------+-------+------------------------------+
| 2025-01-23 16:30:00 |  1001 | quarterly_report.pdf         |
| 2025-01-23 16:32:00 |  1002 | budget_projection_2025.xlsx  |
| 2025-01-23 16:34:00 |  1003 | client_presentation.pptx     |
| 2025-01-23 16:36:00 |  1004 | employee_handbook_2025.pdf   |
| 2025-01-23 16:38:00 |  1005 | investment_analysis.docx     |
| 2025-01-23 16:40:00 |  1006 | financial_report_2024.pdf    |
| 2025-01-23 16:42:00 |  1007 | project_plan_Q1_2025.xlsx    |
| 2025-01-23 16:44:00 |  1008 | HR_policies_2025.pdf         |
| 2025-01-23 16:46:00 |  1009 | IT_security_policies.pdf     |
| 2025-01-23 16:48:00 |  1010 | compliance_audit_report.pdf  |
+-------------------+-------+------------------------------+
```

Feature Engineering: Example 2

Raw Data Field	Derived Feature Name	Description	Missing Val Reason	Imputation Strategy
user_id, time	login_count	Num of logins for a user on day		
user_id, time	login_30avg	login_count / avg logins per day for last 30 days for user	Denominator is Null: last 30 days no logins	Impute denominator with 1
attachment				
time				

Goal:	Anomaly Detection
Data for the model build:	historical records
Time frame for training:	?
Time frame for testing:	?

```
root
 |-- time: string (nullable = true)
 |-- user_id: integer (nullable = true)
 |-- attachment: string (nullable = true)
```

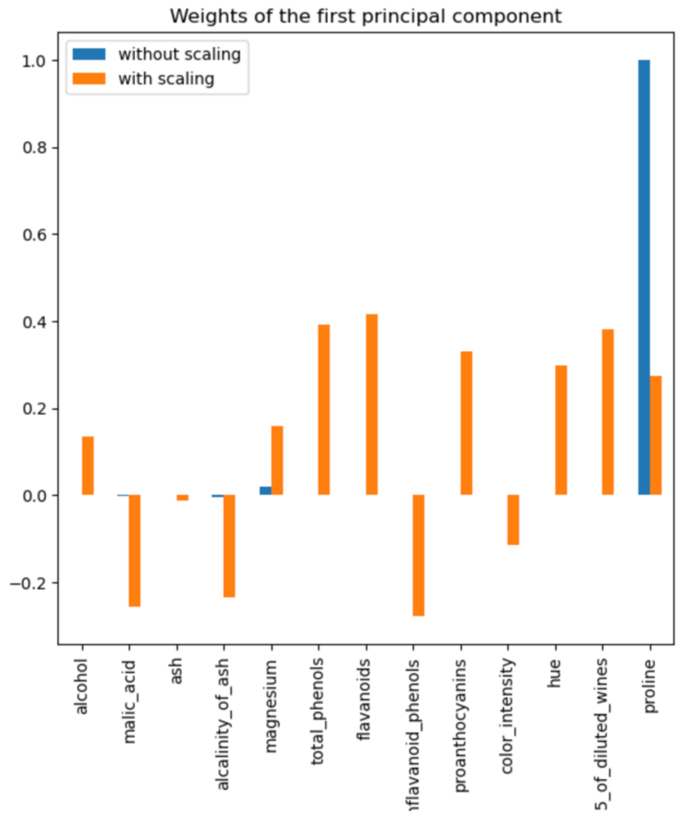
```
+-------------------+-------+------------------------------+
| time              |user_id| attachment                   |
+-------------------+-------+------------------------------+
| 2025-01-23 16:30:00 |  1001 | quarterly_report.pdf         |
| 2025-01-23 16:32:00 |  1002 | budget_projection_2025.xlsx  |
| 2025-01-23 16:34:00 |  1003 | client_presentation.pptx     |
| 2025-01-23 16:36:00 |  1004 | employee_handbook_2025.pdf   |
| 2025-01-23 16:38:00 |  1005 | investment_analysis.docx     |
| 2025-01-23 16:40:00 |  1006 | financial_report_2024.pdf    |
| 2025-01-23 16:42:00 |  1007 | project_plan_Q1_2025.xlsx    |
| 2025-01-23 16:44:00 |  1008 | HR_policies_2025.pdf         |
| 2025-01-23 16:46:00 |  1009 | IT_security_policies.pdf     |
| 2025-01-23 16:48:00 |  1010 | compliance_audit_report.pdf  |
+-------------------+-------+------------------------------+
```

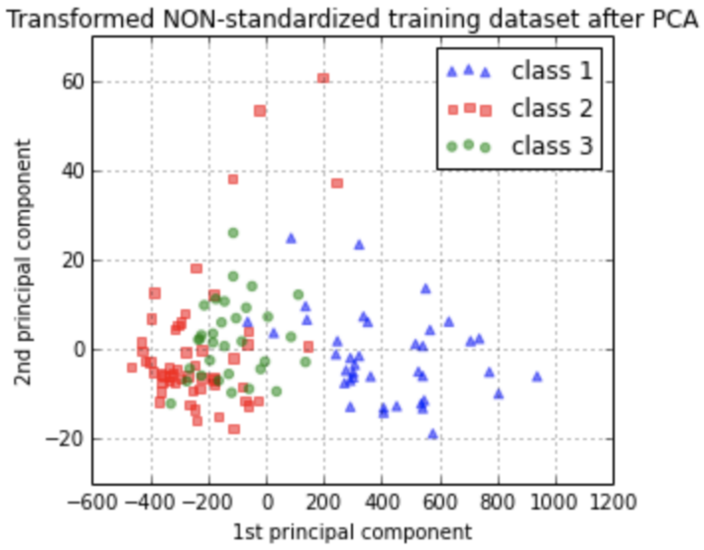
Feature Scaling

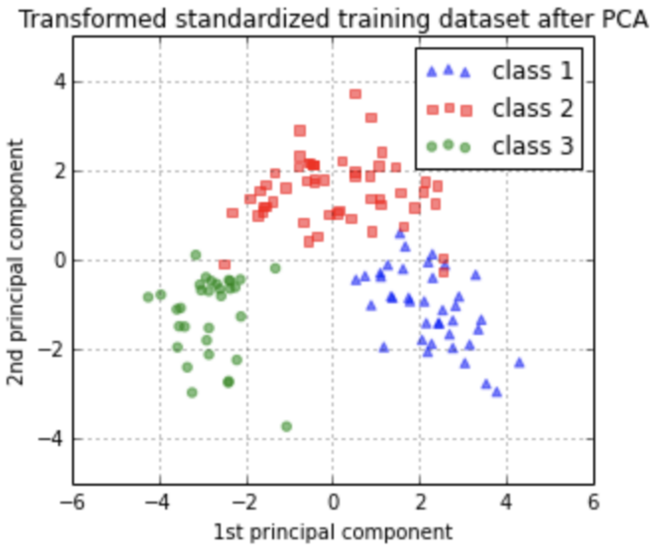
Feature Scaling: PCA

- PCA
  - Reduce dimensions
  - Find features that maximize variance
  - One feature varies more vs others because of their scales, PCA deems feature dominant for direction of the principle component
- Scaling
  - Improves class separability
    - Better model performance

  - https://scikit-learn.org/stable/auto_examples/preprocessing/plot_scaling_importance.html





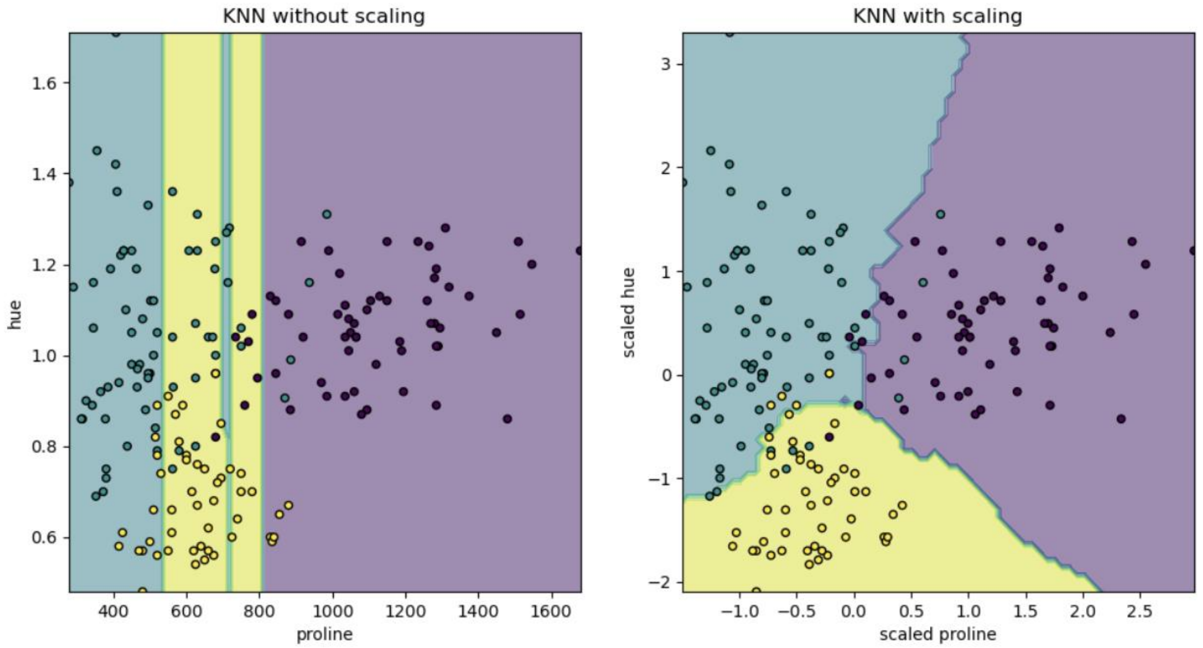


Raschka, S., & Mirjalili, V. (2019). Python machine learning : machine learning and deep learning with python, scikit-learn, and tensorflow 2 / Sebastian Raschka, Vahid Mirjalili. (Third edition.). Packt.

Feature Scaling: k-neighbor models

- Ex: subset of 2 features that have values with different orders of magnitude
  - “proline” values: 0 - 1,000
  - “hue” values: 1 – 10
  - distances between samples are mostly impacted by differences found in the “proline” feature, while values of the “hue” will be comparatively ignored
- Scaling
  - both scaled values lie on approximately the same scale (-3 – 3)
  - neighbor structure will be impacted by both variables

  - https://scikit-learn.org/stable/auto_examples/preprocessing/plot_scaling_importance.html



Ref.: Raschka, S., & Mirjalili, V. (2019). Python machine learning : machine learning and deep learning with python, scikit-learn, and tensorflow 2 / Sebastian Raschka, Vahid Mirjalili. (Third edition.). Packt.

Feature Scaling: Common Approaches

Standardization

- Get features to look like standard normally distributed data:
  - zero mean & unit variance

$$x_{std}^{(i)} = \frac{x^{(i)} - \mu_x}{\sigma_x}$$

- Use:
- many ML estimators assume features have vals close to zero/ vary on comparable scales
  - Unscaled data can slow down/ prevent convergence of many gradient-based estimators
  - *Exception: decision tree-based estimators

Normalization

- rescaling of features to a range of [0, 1]
  - special case of **min-max scaling**
    - apply the min-max scaling to each feature column

$$x_{norm}^{(i)} = \frac{x^{(i)} - x_{min}}{x_{max} - x_{min}}$$

- Use:
- need values in a bounded interval
  - image processing: pixel intensities have to be normalized to fit within a certain range (ex. consider: 0 to 255 for the RGB color range)
  - typical neural network algorithms require data that's on a 0-1 scale

Refer to Raschka for more info:
https://learning.oreilly.com/library/view/python-machine-learning/9781789955750/Text/Chapter_4.xhtml#_idParaDest-88
https://sebastianraschka.com/Articles/2014_about_feature_scaling.html

Scaling: Implementation

NumPy

```
import numpy as np

# Standardization

x_np = np.asarray(x)
z_scores_np = (x_np - x_np.mean()) / x_np.std()

# Min-Max scaling

np_minmax = (x_np - x_np.min()) / (x_np.max() - x_np.min())
```

```
>>> from sklearn.preprocessing import StandardScaler
>>> stdsc = StandardScaler()
>>> X_train_std = stdsc.fit_transform(X_train)
>>> X_test_std = stdsc.transform(X_test)
```

```
>>> from sklearn.preprocessing import MinMaxScaler
>>> mms = MinMaxScaler()
>>> X_train_norm = mms.fit_transform(X_train)
>>> X_test_norm = mms.transform(X_test)
```

Refer to the chapter here:
https://learning.oreilly.com/library/view/python-machine-learning/9781789955750/Text/Chapter_4.xhtml#_idParaDest-89

Corresponding Git:
https://github.com/rasbt/stat479-machine-learning-fs19/blob/master/05_preprocessing-and-sklearn/code/05-preprocessing-and-sklearn__notes.ipynb

Scaling: Implementation

NumPy

```
import numpy as np

# Standardization

x_np = np.asarray(x)
z_scores_np = (x_np - x_np.mean()) / x_np.std()

# Min-Max scaling

np_minmax = (x_np - x_np.min()) / (x_np.max() - x_np.min())
```

```
>>> from sklearn.preprocessing import StandardScaler
>>> stdsc = StandardScaler()
>>> X_train_std = stdsc.fit_transform(X_train)
>>> X_test_std = stdsc.transform(X_test)
```

```
>>> from sklearn.preprocessing import MinMaxScaler
>>> mms = MinMaxScaler()
>>> X_train_norm = mms.fit_transform(X_train)
>>> X_test_norm = mms.transform(X_test)
```

```
mu, sigma = X_train.mean(axis=0), X_train.std(axis=0)

X_train_std = (X_train - mu) / sigma
X_valid_std = (X_valid - mu) / sigma
X_test_std = (X_test - mu) / sigma
```

https://learning.oreilly.com/library/view/python-machine-learning/9781789955750/Text/Chapter_4.xhtml#_idParaDest-89
https://github.com/rasbt/stat479-machine-learning-fs19/blob/master/05_preprocessing-and-sklearn/code/05-preprocessing-and-sklearn__notes.ipynb

Data Preprocessing: Categorical Values & Missing Data

- Notebook:

https://github.com/rasbt/python-machine-learning-book-3rd-edition/blob/master/ch04/ch04.ipynb

- Reference Chapter:

https://learning.oreilly.com/library/view/python-machine-learning/9781789955750/Text/Chapter_4.xhtml#_idParaDest-79