A Machine Learning Primer

Can we build a machine learning model to help us in protein engineering?

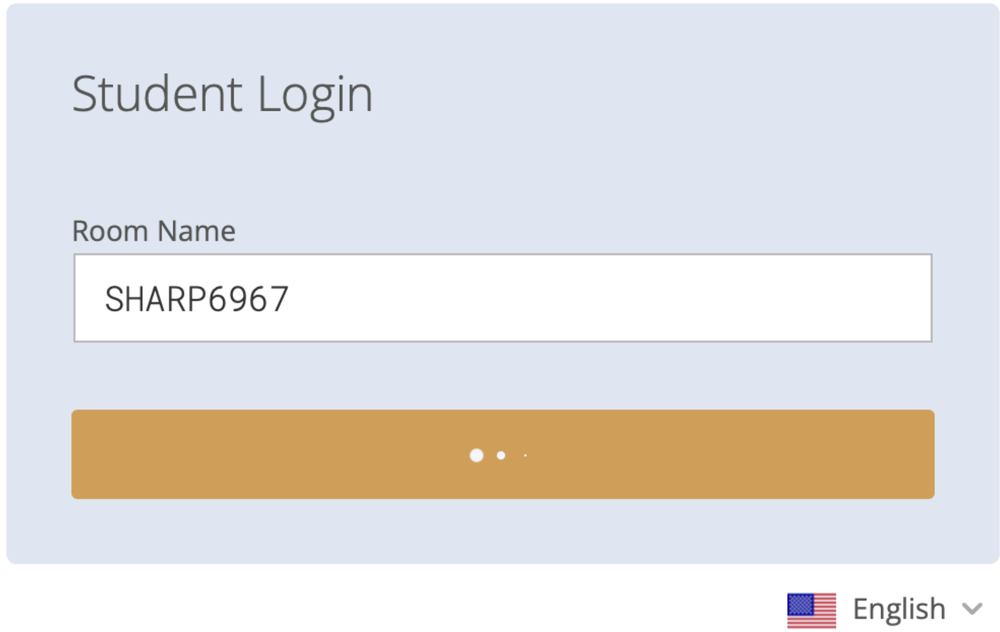
Dr. Sharp **12/15/2022**

Access to Presentation and Hands on Example

https://github.com/lms464/ML_example

Using Socrative

https://b.socrative.com/login/student/



If a bullet point is in **bold** it is a question I encourage you to answer on Socrative

Table of Contents

Goals

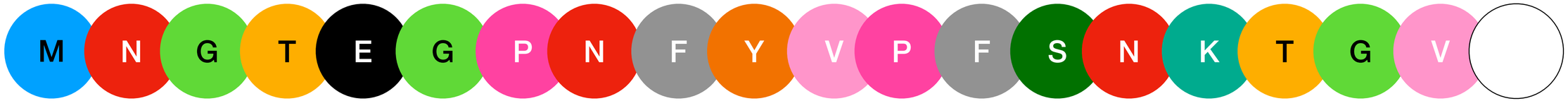
- My goal is for you to take home an introductory understanding of
  - What is machine learning?
  - How can it be applied to proteins?
  - Explanation of the machine learning hands on example

Table of Contents

What are proteins

- From your classes you know protein sequence → structure

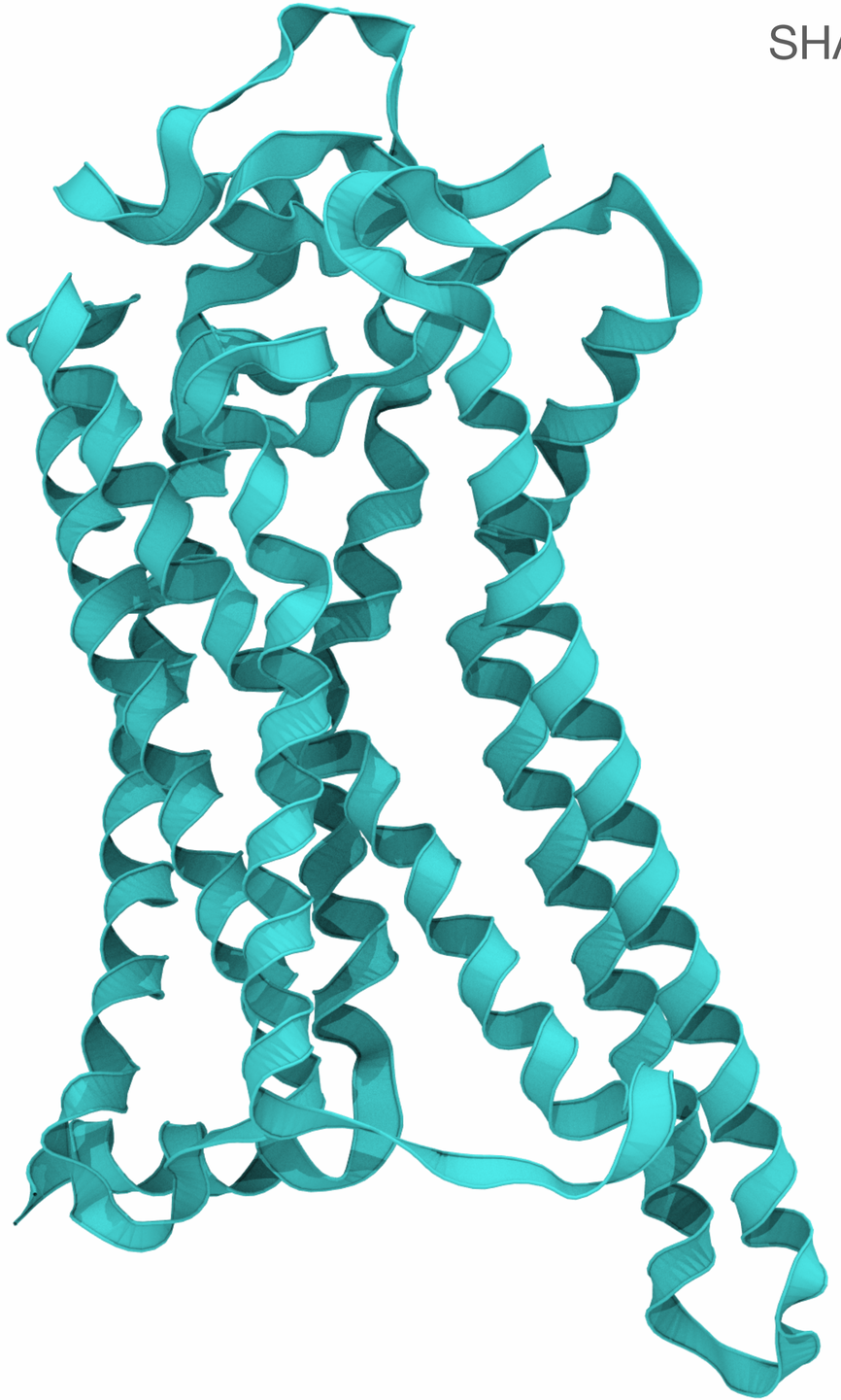
Rhodopsin



What are proteins

- From your classes you know protein sequence → structure

Rhodopsin

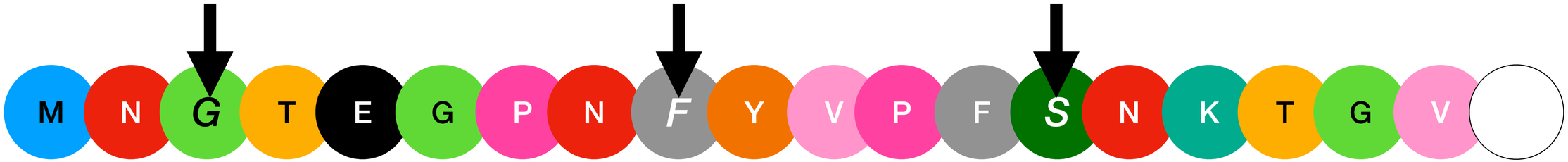


- Structure → function

- If the sequence is changed (a mutation) it may
  - Improve function
  - Inhibit function
  - Do nothing

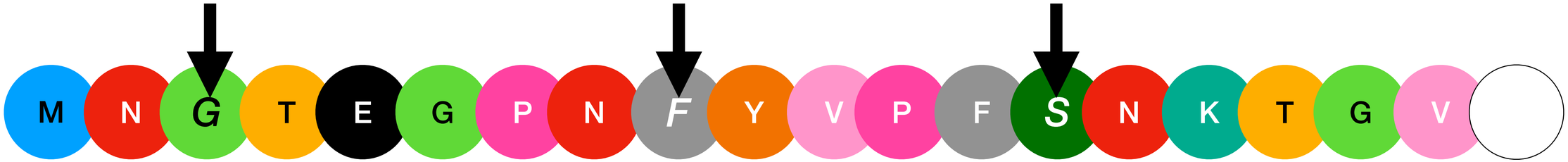
What are proteins: a thought experiment

- Pretend our hypothetical protein always folds right and is always found in it’s optimal environment
- Say we have 3 locations we can mutate



What are proteins: a thought experiment

- Pretend our hypothetical protein always folds right and is always found in it's optimal environment
- Say we have 3 locations we can mutate



- **How many combinations are there? (3 mutation points 20 amino acids)**
    A. Less than 1000
    B. More than 1000

What is machine learning

ML is a family of algorithms — it analyzes and trains on large data sets to make predictions and find patterns

Machine Learning Algorithm

https://prateekvjoshi.com/2013/01/03/can-machines-be-truly-independent/thinking-computer/

How is ML used in daily life

- **How might machine learning be used in your every day life?**
- Google searches learn
- Siri learns langues and how to convert them to text or commands
- CAPTCHA

How is ML used in biology

- Health care - predict patients illnesses
- Ligand/Drug binding - What small molecule will fit in a binding pocket with the highest affinity
- Protein folding* - How do we take a sequence and predict structure
- Directed Evolution - How can we predict a mutation to a amino acid sequence that improves a protein’s function
- Plenty more

*__NOTE__: Not actually ML, more a cousin
Called Deep Learning and tries to mimic how a brain works.

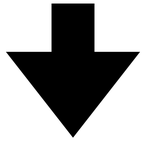
Introduction Review

- Proteins are a sequence of amino acids - their structure and sequence determine function
    - Mutations may adjust this function
- Machine learning is a process of teaching an algorithm to find patterns and make predictions
- Machine learning is found in your daily life and used to solve complex biological questions

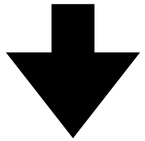
Table of Contents

Our Machine Learning Pipeline

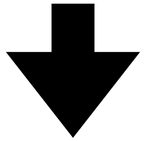
What problem do you want to understand?



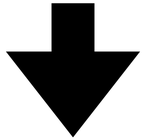
Clean your data



Choose a ML algorithm to train with



Train and evaluate the algorithm on data



Tune and test

Our Problem

- This paper looks at channelrhodopsin — a light gated ion channel
    - Different colors of light produce different currents
    - Or different features
- The authors wanted to predict a new sequence of channelrhodopsin to activate in lower light

ARTICLES
https://doi.org/10.1038/s41592-019-0583-8
nature methods

Machine learning-guided channelrhodopsin engineering enables minimally invasive optogenetics

Claire N. Bedbrook[1], Kevin K. Yang[2,3], J. Elliott Robinson[1,3], Elisha D. Mackey[1], Viviana Gradinaru[1*] and Frances H. Arnold[1,2*]

Data

What kind of data do we need

- First and foremost: ground truth data
    - You need a reference sequence and quantified data on functionality
- You need a lot of data
    - How much data is dependent on your project
- This data should also be multivariable (have multiple features)
- Some potential features:
    - Current produced from a specific color
    - Sequence
- These features will be used to determine a prediction

5 petabytes of black hole data

Data

Cleaning Data

- Data must be in a particular format
- Use a table with multiple columns
- Each column is a feature — the current
- Each row is a specific protein
- Deal with bad data:
    - Empty index - NAN
    - Repeated data — remove
    - Varying format — uniform the data

Data

An example of data

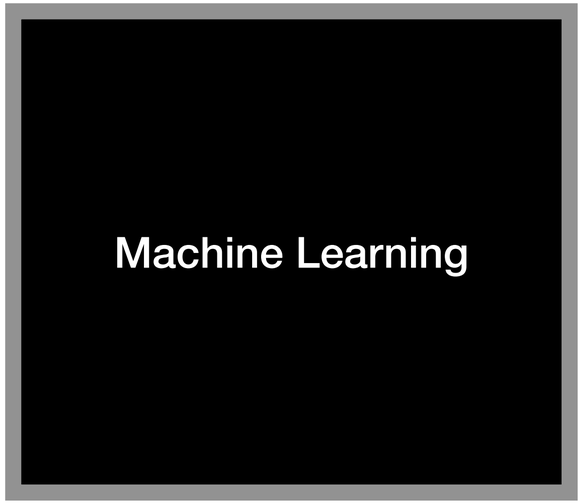
41592_2019_583_MOESM5_ESM

ChR_name	cyan_peak (nA)	green_peak (nA)	red_peak (nA)	cyan_ss (nA)	green_ss (nA)	red_ss (nA)	kinetics_off	max_peak (nA)	max_ss (nA)	max_ss (nA)	Amino_acid_sequence
C1C2	0.66	0.16	0.01	0.45	0.14	0	28	0.66	0.45	0.45	MSRRPWLLALALAVALAAGSAGAA
c1	0.14	0.01	0.01	0.08	0	0	7	0.14	0.08	0.08	MSRRPWLLALALAVALAAGSAGAA
CsChrim	0.83	0.98	0.77	0.69	0.77	0.42	51	0.98	0.77	0.77	MSRLVAASWLLALLLCGITSTTTASA
CheRiff	0.66	0.06	0.01	0.46	0.05	0	16	0.66	0.46	0.46	MGGAPAPDAHSAPPGNDSAAHIVM
c62	0.01	0.01	0.01	NAN	Zero	0	NAN	0.01	0	0	MSRLVAASWLLALLLCGITSTTTASA
c64	0.48	0.7	0.41	0.42	0.6	0.27	61	0.7	0.6	0.6	MSRLVAASWLLALLLCGITSTTTASA
c64	0.48	0.7	0.41	0.42	0.6	0.27	61	0.7	0.6	0.6	MSRLVAASWLLALLLCGITSTTTASA

What miss handled data do you see?

Choosing an approach to train our algorithm

- How do we teach our algorithm to classify and predict a new functionally relevant sequence?
- Broadly - 2 types of approaches (there are more)
    - Supervised
    - Unsupervised



Choosing an approach to train our algorithm

Supervised

- **Supervised** - The algorithm is provided data and a solution and learns how to relate the two together.
  - This is you practicing for a math exam by correcting your work with answers
  - Used in classification and ranking
  - Every day example: **spam filters**

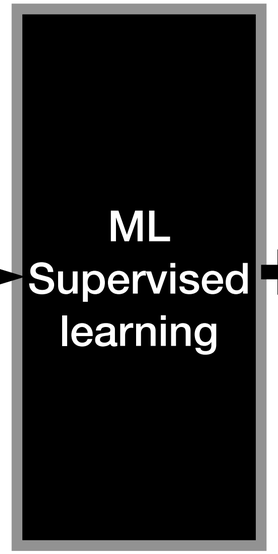
Choosing an approach to train our algorithm

Supervised Learning “Example” - Predicting sequences from current

- With the data we have seen, how might this look?

Features - Describing a sequence

ChR_name	cyan_peak (nA)	green_peak (nA)	red_peak (nA)	cyan_ss (nA)	green_ss (nA)	red_ss (nA)	kinetics_off	max_peak (nA)	max_ss (nA)
C1C2	0.66	0.16	0.01	0.45	0.14	0	28	0.66	0.45
c1	0.14	0.01	0.01	0.08	0	0	7	0.14	0.08
CsChrim	0.83	0.98	0.77	0.69	0.77	0.42	51	0.98	0.77
CheRiff	0.66	0.06	0.01	0.46	0.05	0	16	0.66	0.46
c62	0.01	0.01	0.01	NAN	0	0	NAN	0.01	0
c64	0.48	0.7	0.41	0.42	0.6	0.27	61	0.7	0.6



Labels - The predictions we are training for

Amino_acid_sequence
MSRRPWLLALALAVALAAGSAGA
MSRRPWLLALALAVALAAGSAGA
MSRLVAASWLLALLLCGITSTTTAS
MGGAPAPDAHSAPPGNDSAAHIV
MSRLVAASWLLALLLCGITSTTTAS
MSRLVAASWLLALLLCGITSTTTAS

Choosing an approach to train our algorithm

Unsupervised

- **Unsupervised** - The model is provided data and works out the patterns
  - Student trying to go through a class without any help
  - Often used in clustering
    - Or how do we divide our data into groups
  - Every day example: **predicting how customers buy**

Choosing an approach to train our algorithm

Supervised and Unsupervised

- **Supervised** - Features and labels provided. Used to classify or rank data
  - **Pro**: generates specific and tailored models based on provided data
  - **Con:** requires large amount of data with features and labels and evaluating often is done by hand
- **Unsupervised** - Only features provided. Used to cluster data
  - **Pro:** No need for labeled data and recognition happened “automatically”
  - **Con**: Output may not use data in a relevant way

Choosing an approach to train our algorithm

Supervised and Unsupervised

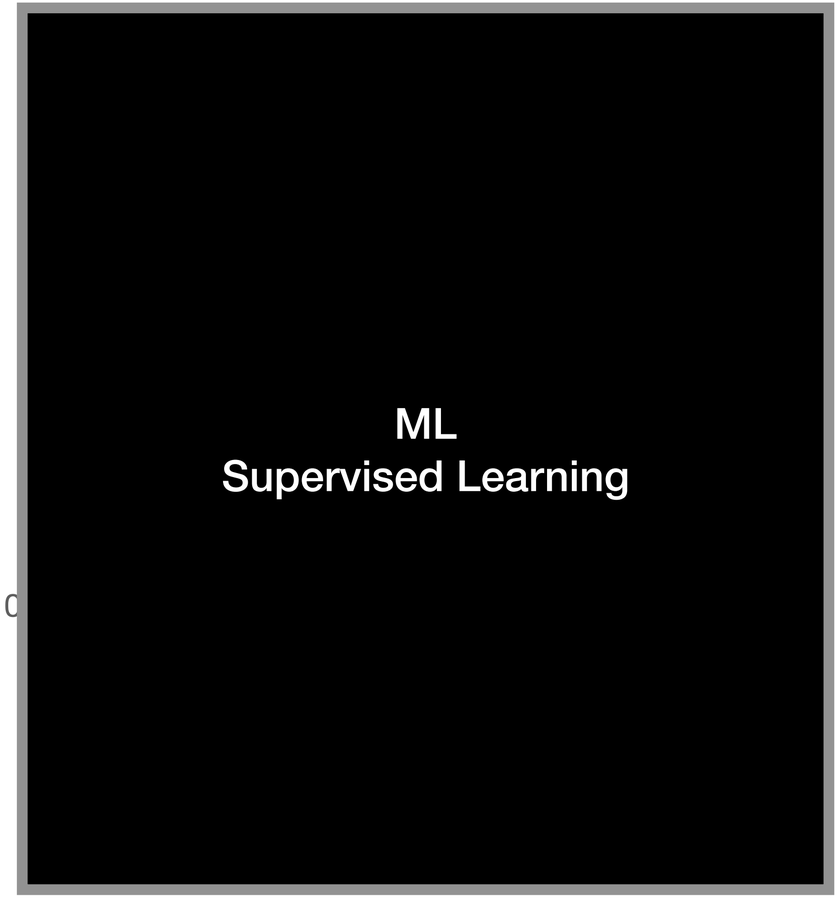
- **Take a minute and discuss the differences between supervised and unsupervised learning and put your answers in Socrative**

Choosing an approach to train our algorithm

Supervised learning options

- What is the ML black box?
- There are numerous algorithms we can use

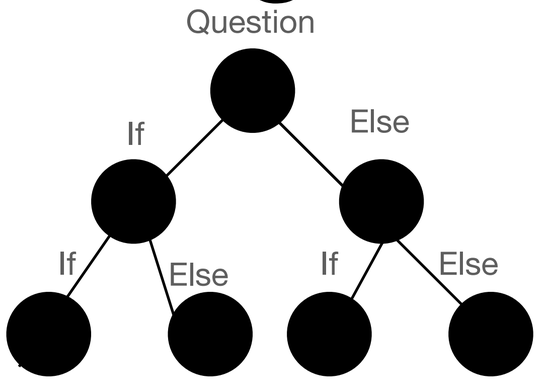
https://doi.org/10.10

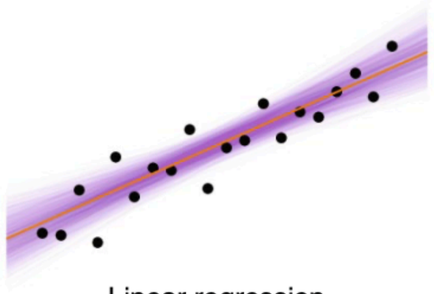


Choosing an approach to train our algorithm

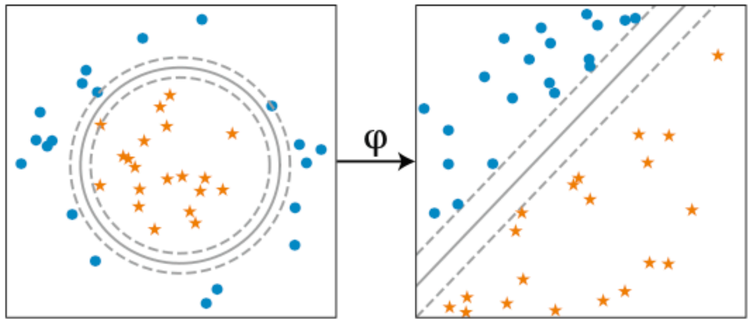
Supervised learning options

- What is the ML black box?
- There are numerous algorithms we can use
- Some supervised models are:
    - Linear regressions - Finding a linear line that fits our data. You have used this plotting noises data in lab
    - Support Vector - finding a line or curve that bests groups our data. Used in image analysis
    - Decision Trees - filtering data through a series of questions. Similar to multiple if-else statements to make divisions
    - K Nearest Neighbors - classify data by observing how the data around it is classified. Powerful tool to make predictions
- There are costs and benefits to each model
- Also note: it is possible to use more than one approach



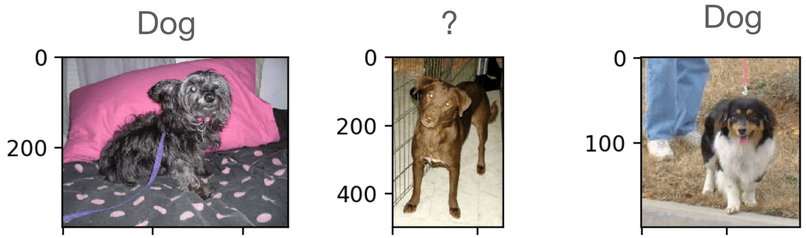


Linear regression



Support vector machine

https://doi.org/10.1038/s41592-019-0496-6



Choosing an approach to train our algorithm

What has been used in directed evolution

- The nature paper uses gaussian processing (classify) — useful for finding trends in noisy data then regressions (predict)

ARTICLES
https://doi.org/10.1038/s41592-019-0583-8
nature methods

Machine learning-guided channelrhodopsin engineering enables minimally invasive optogenetics

Claire N. Bedbrook[1], Kevin K. Yang[2,3], J. Elliott Robinson[1,3], Elisha D. Mackey[1], Viviana Gradinaru[1*] and Frances H. Arnold[1,2*]

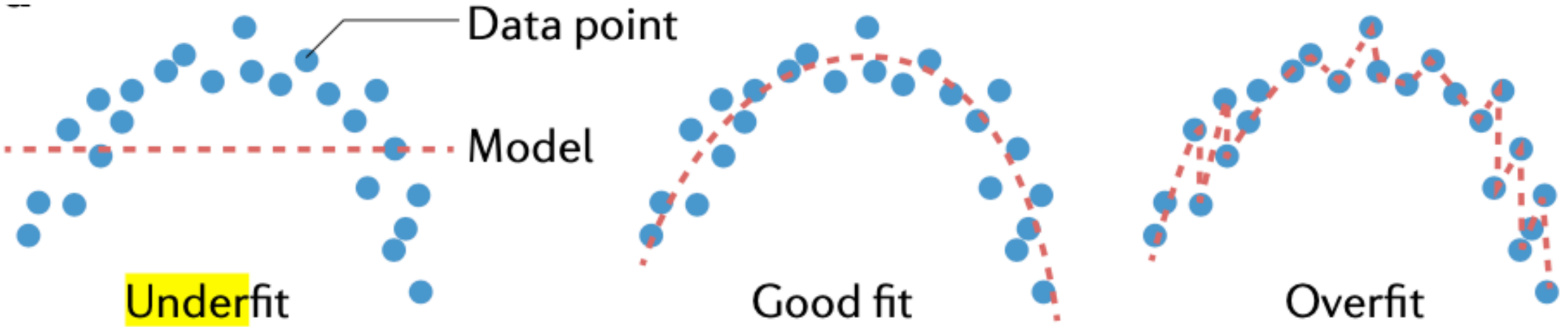
Train the model

- We have access to python and R libraries that handle the actually training, scikit.learn is used here
- We want to split up our data into a training set and a testing set
- Use your training set of data first!
- This is your algorithm going to class

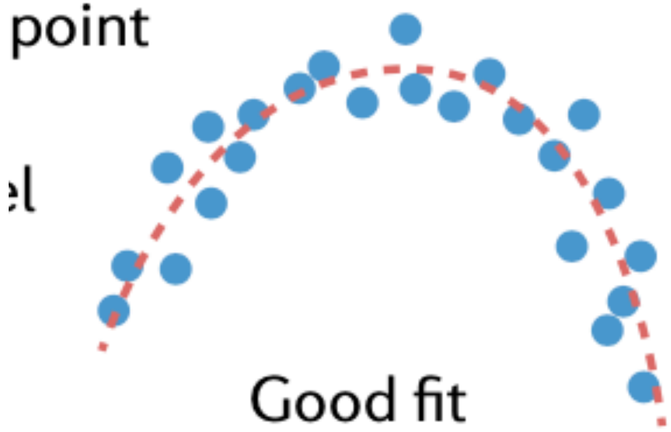
Evaluate the model

How well does the model fit to the data?

- If training your model is like going to class, evaluating your model is like taking an exam

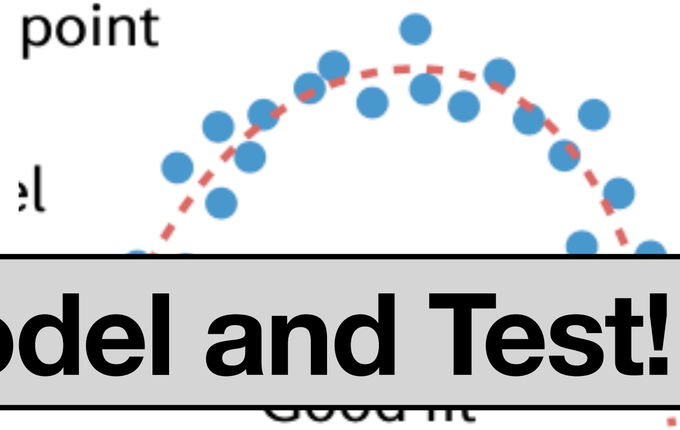


Tuning the model



- After we go a round of training we tune hyper-parameters
- Hyper-parameters do not adjust data only how you model learns
  - Learning rate — how fast you let your algorithm learn
  - Iterations used — how many time you've trained your model
  - How you have split your data - 70:30 Train:test; 50:50 Train:Test?
- Once you have tuned your model, run your test set and confirm if works

Tuning the model



Tune your model and Test!

- After we go a round of training we tune hyper-parameters
- Hyper-parameters do not adjust data only how you model learns
  - Learning rate — how fast you let your algorithm learn
  - Iterations used — how many time you've trained your model
  - How you have split your data - 70:30 Train:test; 50:50 Train:Test?
- Once you have tuned your model, run your test set and confirm if works

Predictions!

Photocurrent	Color Peak	k_off	Generation
1.2	0.001	0	3
2.0	1	2.3	1
4.2	0.09	4.3	2

New Mutation!
?
?
?

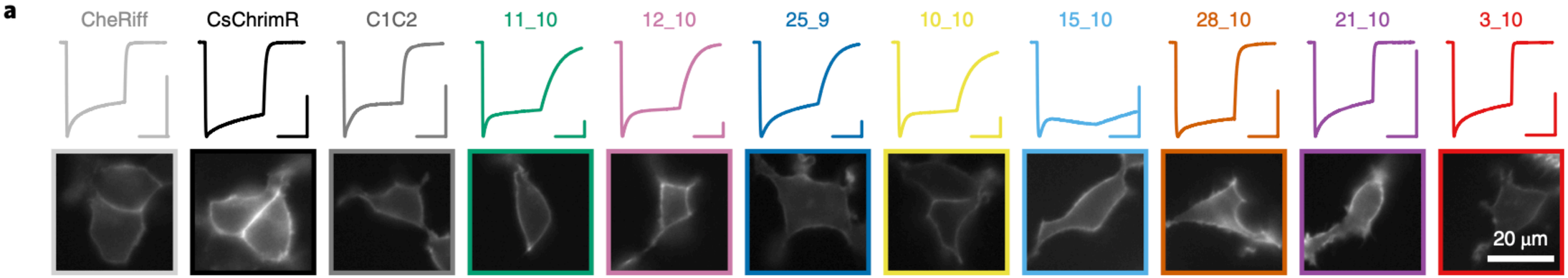
Predictions!

https://prateekvjoshi.com/2013/01/03/can-machines-be-truly-independent/thinking-computer/

ARTICLES
https://doi.org/10.1038/s41592-019-0583-8
nature methods

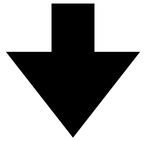
Machine learning-guided channelrhodopsin engineering enables minimally invasive optogenetics

Claire N. Bedbrook[1], Kevin K. Yang[2,3], J. Elliott Robinson[1,3], Elisha D. Mackey[1], Viviana Gradinaru[1*] and Frances H. Arnold[1,2*]

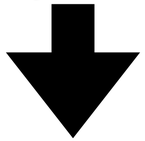


Predictions!

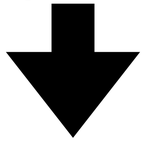
What problem do you want to understand?



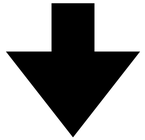
Clean your data



Choose a ML algorithm to train with



Train and evaluate the algorithm on data



Tune and test

Table of Contents

Access to Example

- We do not have time to build or train a model for protein
- This link will take you to an example training an algorithm to find predict and classify flowers

To use follow steps in README.md

How the Jupyter Notebook Looks

In [8]:
```
# X contains all features treated as through they describe the dropped feature
# y is the label we are training for

X = data.drop(["variety"],axis=1)
y = data["variety"]

# consider what the shapes are
# are X and y a matrix or a vector?
print(X.shape)
print(y.shape)
```

```
(150, 4)
(150,)
```

In [23]:
```
# experimenting with different k values
# k is a hyperparameter, tuning for a specific k is important

k_range = list(range(1,50))
scores = []
for k in k_range:
    # model we are using is Kneighbors - used for classificaiton
    # n_neighbors -> setting the number of neighbors to compare

    knn = KNeighborsClassifier(n_neighbors=k)

    # fit evaluates our model and is our first line of validation
    knn.fit(X, y)

    # check how well your model predicts your data
    y_pred = knn.predict(X)

    # quantify your results, determines the number of accuratly predicted values
    scores.append(metrics.accuracy_score(y, y_pred))

plt.plot(k_range, scores)
plt.xlabel('Value of k for KNN')
plt.ylabel('Accuracy Score')
plt.title('Accuracy Scores for Values of k of k-Nearest-Neighbors')
plt.show()
# How does your accuracy varry with number of neighbors? What are the best values of k?
```