Week 4 – day 1 - Python Data science

Pandas – This is like dataframes. Built in numpy – matrices of numerical data

Matplotlib – plotting but extensive package

Seaborn instead of matplotlib

Jupyter lab – like Rmd

jupyter lab --no-browser --port=5007

<http://localhost:5007/?token=05e82dde7beccb5d4c0cdf5b3d92b67c966a668197b170a2>

Alter config to add nb: slide 4

ProxyCommand and ssh should be same line

Then ssh nb. Paste the link above into firefox

Next we’re gonna do Numpy –

Panada built on numpy arrays – allows you to add metadata to rows n columns

Label based indexing – easy handling of data – missing data

Tidyverse – formatting the data

Series is a one dim

#TwoD – rows labels etc

Join and merges – different ways to join tables (but if they have a shared column etc)

Specify key/table etc

Join – merging on an index rather than value in table

Grouping – splitting into groups based on genes, apply an independent function. Apply, combine, reduction

Reshaping – like tidydata – reshaping/melting the data to fit into a specified structure, pivoting.

Plotting library – matplotlib – how we’re going to ultimately plot the data

We’re instead going to use seaborn – works well with pandas/tidy data. Rug tells you where the actual points are on axis

Count data from week 1- copy it to juptyer folder, load it and start manipulating etc

Counts.rename(columns={’previous’:’new’}, inPlace=True)

Git remote -v

Morning of Tuesday 19th – going through the code from yesterday

• Add TAD information to dataframe (http://chromosome.sdsc.edu/mouse/hi-c/download.html)

• Calculate average expression for each TAD

#Lecture – linear regression

X is independent, y is dependent

B0 is c b1 is m

Pearsons – correlation between x n y (r)

Should only reall use linear regressions for non-perfect relationships. Shouldn’t use it for perfect/deterministic data – would just get the gradient of the line

Closed form/convex – only one solution with a line

Ransack- fits bulk of data and ignores outliers, iterative – bootstrap – as linear regression is sensitive to outliers

If your independent outliers are related

Zip is when you have multiple arrays of same length – when you want of each at each interval – basically elementwise way of doing something across vectors

x1 = [1, 2, 3]  
x2 = ['a', 'b', 'c']  
x3 = ['x', 'y', 'z']

for i1, i2, i3 in zip(x1, x2, x3):  
    print(f"x1: {i1}\tx2: {i2}\tx3: {i3}")

Will print this:

x1: 1    x2: a    x3: x  
x1: 2    x2: b    x3: y  
x1: 3    x2: c    x3: z

How to check the robustness of the model: square residuals and sum them up -if this value is very high then could questin the robustness of the model

Is your data non linear etc?- importan to visualise the distribution

<https://github.com/dwaithe/linear_regression_practical>

Dimensionality reduction:

Swiss roll dataset – given v native dimensionality of data

Volume v

X – samples

Y – labels

N – number of samples

D- dimensionality of samples

D – desired dimensionality

Edge weight – node or vertex(points) we want to connect them to 4 nearest neigbours- connection becomes an edge- has a weight, min hops to get from a to b – some edges longer so bigger weight

tSNE/UMAP – connect each node to (k) nearest neighbours – how do you choose k? set an expected number of neightbours but don’t enforce

compute distances along graph in a probalistic way – forces between nodes – if force Is strong between points then points cluster together. Position the points in a lower dimensional space

absolute distances mean nothing with tSNE or UMAP – samples that are close in low ds are close in higher. But samples that are far aart in low ds may be close in high ds

Work better than isomap – look at adjacent neighbours and set a seed/stochastic random forces

Isomap suffers from local conc of forces (defined streaks on the plot)

Clusters:

Attribution to archetypes

Neighbour distances

This is all unsupervised – what about when you have a set of labels etc

Targeted dimensionality reduction while preserving max amount about targeting – linear discriminatory analysis

Preserving specific aspects of your data – related algorithm

Paul B would recommend linear discriminant analysis (skikit learn) Assumes data is linear

Labelled variants of UMAP – merge datasets single cell RNAseq/citeseq together

Port ssh -N -f -L localhost:5000:localhost:5000 h1

Thursday – Machine Learning with Ed Morrissey

Kaggle – predictive modelling/competition – way to make money etc and also to learn

Scikit learn (python library) nicely structured, documented etc – makes machine learning more accessible

Training data – predictions (supervised is giving a training set – known I supposed)

Derive a set of rules from known dataset

Unsupervised – don’t know what you’re looking for, i.e clustering – dimensionality reduction etc.

Iris dataset – using this model to train a dataset to predict species ffrom a dataset of iris

Sklearn (load iris)

Classification – partition space into regions so if a point falls into one region you call it one thing. Kernel – different effct on how lines are drawn (radial is more circular) partitions the space differently

SVM is partitioning space.

Decisions trees (simple and popular) sequence of yes no decisions. Sequence of is this variable bigger than this – series of partitions. Ie. Is CD4 expression on ? If yes then – next. Set of rules, simplistic. Overfit – doesn’t predict well outside of training data

Sklearn.tree DecisionTreeClassifier

Clf = clf.fit(data, target) #create a classifier, derives a set of rules to decide rules between different types.

Clf.predict(data[:2, :]) predicts on first 2 lines

Overfitting refers to overtraining on the data that you’ve given it so can’t predict on new data

If you partition your data (test v train) can see if it can predict – see if it overfits

Train test split into a test size of 0.25 –

Fit only to training data then try to fit test data – predicition not as good as before.

Evaluate how much you’re overfitting.

Methods to minimise this (random forest) – using decisions trees in a particular way

Decision trees on subsets on data – bunch of trees randomly, don’t use all variables at one. Splitting it into chunks, forests of trees so use the chunks together to reduce overfitting

Clf\_forest.score

Scikit learn dataset – loads of different models – effect of different classifiers and their performance

How do you decide which to use?

Random forest is quick and good – split into training/test etc

Overlap with statistics a lot here

Next: Dimensionality reduction (unsupervised) – uncover strcture and plot it

1. PCA

You get as many PC out as you put in – orders them based on how much variance explained/amount of info

1. tSNE

Calculate distance between cells using gaussian – 2D arrangement that respects distances

1. UMAP

Similar to tSNE (faster) because of intiialise. Can give classes or add new data

Knn – K nearest neighbours – find k closest cells – join up to get an interconnected graph. Subtract expression take sum square them all to give Euclidean distance and find nearest cells.

How do you choose K (5) – if you have small distnt cell pops then better to use fewerk as otherwise will start bringin in cells that it shoulder

Spring/Seurat – force directed layouts. Paths in the data better reproduced

Louvain clustering/Phenograph – networks finds the clusters

Machine learning vs stats

Stats generally more rigorous

ML – good for big data, speedier, less interested in underlying data

ML easier to just have a go

Outliers (supervised) need training dataset that has typical data and variance – if not, then wont extrapolate well to more varied dataset

Deep neural networks

10x genomics – dataset pbmc -2017

Purified different cell populations (beads/antibodies)

10 different populations measured separately

Take these different populations and use them as classifiers

Read files, concatenate, plot, initialise different classfiers – fit/test overfit

Download new data

Friday 22nd Charlotte Rich Griffin - Dendrou

Single cell RNAseq – using scanpy/python

Smart-seq2 v 10x – do you want lots of cells v indepth look at smaller number of cells

Smart-seq2 is plate based

How to integrate repertoire data for T cells with single cell etc

Preprocessing,batch processing, doublets and multiplets - demultiplexing etc

Normalisation – compare cells directly

Identify highly variable genes

Frameworks for single cell analysis – Seurat, singleCellExperiment, scater

Python – scanpy – it can deal with big data sets (better with machine learning – quicker)

Easier to add advanced modelling like scikit-learn/Tensorflow

Scanpy built on anndata class (object)

Hard to switch between packages as each works on it’s own object

API – 3 functions (sc.pp/tl/pl) sc.tl umap but may also have sc.pl umap

Single nuclei RNAseq – own challenges (fat cells – Andrea)

Demuxlet – a way of genetically debarcoding cells if the cell hashing doesn’t work

/ifs/obds-training/apr20/rose/JupyterLab/Scanpy

Regressing for mt content – be careful on scTransform (T cells have a similar mitochondrial contnt so don’t need to worry about affecint a particular subset). Also activated cells will have lower mitochondrial content as they express more genes

Good to be lenient at first then go and look at the clusters and impose counts, mt content etc and see if clusters can be explained by these things – then be harsher and see if they cluster together

Removal of TCR genes so clusters aren’t determined by repertoire, more by their transcriptomic profile

How to decide whether to regress or not – Do dimensionality first (are particular principal components influenced by counts etc? Look on the UMAP

<https://github.com/theislab/scanpy_usage/blob/master/180209_cell_cycle/cell_cycle.ipynb>

Look at contamination

[https://www.biorxiv.org/content/10.1101/303727v2](https://www.biorxiv.org/content/10.1101/303727v2" \o "https://www.biorxiv.org/content/10.1101/303727v2" \t "_blank)

<https://github.com/constantAmateur/SoupX>