Exam 14th of January 2020 for the course 1MS041 (Introduction to Data Science)

1. Fill in your anonymous exam code in the cell below.
2. Complete the Problems by following instructions.
3. When done, submit this file with your solutions saved, as instructed on Studium.

Any questions regarding the exam can be asked by sending a message through Studium to the t
When asking a question about a specific problem, be sure to explain the problem clearly as all e
shuffled in order.

In [0]:
```
# Enter your anonymous exam id by replacing XXXX in this cell below
# do NOT delete this cell
MyAnonymousExamID = 'XXX'
```

PROBLEM 1

Maximum Points = 5

Consider the following matrix $$ M = \begin{bmatrix} 1 & 1 \\ 0 & 3 \\ 3 & 0 \end{bmatrix} $$ Manu
hand, produce the

- [1p] Rank: `rank`
- [1p] Left singular vectors in matrix form: `V`
- [1p] Singular values in matrix form: `D`
- [1p] Right singular vectors in matrix form: `U`

If $UDV^T = M$ then [1p].

When answering these questions, use the sagemath `matrix` format as the example below. Us
answers, i.e. use `sqrt(3)` if that appears in the calculation.

> To make sure that we all agree on signs, make sure that the sign of the first component of each singular vector is positive.

In [36]:
```
M = matrix([[1,1],[0,3],[3,0]])
```

In [ ]:
```
rank = XXX # [1p] The rank of the matrix M
V = XXX # [1p] Matrix of singular vectors, each vector is a column of V
D = XXX # [1p] The square matrix with the singular values on the diagonal
U = XXX # [1p] The matrix of right singular vectors (each vector is a colu

## Hint, use V.nrows() and V.ncols() etc. to make sure that you got the di
## expect. If nrows does not work, then you should make sure you are usin

## [1p] Hint: Check that U*D*V.transpose() == M
```

PROBLEM 2

Maximum Points = 5

Consider the n IID θ-parametric family of rescaled Rademacher random variables with t
following probability mass function:

$$ f(x; \theta) = \begin{cases} \theta & \text{ if } x=+10\\ 1-\theta & \text{ if } x=-10\\ 0 & \text{ othe
\end{cases} $$

Your task now is to perform a Wald Test of size $\alpha=0.05$ to try to reject the null hypothesis
chance of seeing a $+10$ is exactly $1/2$, i.e., $$\displaystyle{H_0: \theta^*=\theta_0 \quad \tex
versus } \quad H_1: \theta^* \neq \theta_0, \qquad \text{ with }\theta_0=0.5}$$ Show you work by
replacing `XXX`s with the right expressions in the cell below.

In [ ]:
```
dataSamples2 = np.array([-10,+10,+10,-10,-10,-10,+10,+10,-10,-10,-10,+10,-
# HINT: to get the counts of +10s and -10s, perhaps a useful statistic fo
print (sum(dataSamples2==-10)) # number of -10s
print (sum(dataSamples2==+10)) # number of +10s
```

In [ ]:
```
## HINT: Think how the likelihood here is related to that for Bernoulli t

## STEP 1: get the MLE thetaHat,
### either by hand or numerically
thetaHat=XXX
print ("mle thetaHat = ",thetaHat)

## STEP 2: get the NullTheta or theta0
NullTheta=XXX
print ("Null value of theta under H0 = ", NullTheta)

## STEP 3: get estimated standard error
seTheta=XXX # recall standard error comes from Fisher Information
print ("estimated standard error",seTheta)

# STEP 4: get Wald Statistic
W=XXX
print ("Wald statistic = ",W)

# STEP 5: conduct the size alpha=0.05 Wald test
# do NOT change anything below
rejectNull1 = abs(W) > 2.0 # alpha=0.05, so z_{alpha/2} =1.96 approx=2.0
if (rejectNull1):
    print ("we reject the null hypothesis that theta_0=0.5")
else:
    print ("we fail to reject the null hypothesis that theta_0=0.5")
```

PROBLEM 3

Maximum Points = 5

In the next cells the earthquake data is analyzed further, specifically depth and magnitude.

Your task is to understand the analysis and continue with the following:

1. Make a residual plot for the fit and discuss briefly the scatter of residuals. Explain what value are farthest from the x-axis (both below and above) mean here.
2. Conduct a Wald test with alpha at 5% of the null hypothesis that $\beta_1=0$. State your co by setting the Boolean variable `RejectNullHypothesisForProblem4` to `True` if you and `False` if you do not.

In [2]:
```
# Lets extract the depth and magnitude from myProcessedList
# (which contains: longitude, latitude, magnitude, depth and the origin t
eqData = np.array(myProcessedList)[:,[3,2]]
eqDepth = eqData[:,0]
eqMagnitude = eqData[:,1]
```

In [ ]:
```
from scipy.linalg import lstsq
import matplotlib.pyplot as plt
import numpy as np

M1 = eqDepth[:, np.newaxis]^[0, 1]
b, res, rnk, s = lstsq(M1, eqMagnitude)
plt.plot(eqDepth, eqMagnitude, 'o', label='data')

xx = np.linspace(0.0, 550, 101)
yy = b[0] + b[1]*xx
plt.plot(xx, yy, label='least squares fit')
plt.xlabel('Earthquake Depth (X)')
plt.ylabel('Earthquake Magnitude (Y)')
plt.legend(framealpha=1, shadow=True)
plt.grid(alpha=0.25)
plt.text(3, 8.5, r'$\widehat{r}(x) = \widehat{\beta}_0 + \widehat{\beta}_1
\widehat{\beta}_0 = $ %(b0)0.3f , $\widehat{\beta}_1 = $ %(b1)0.3f' % {'b(
plt.show()
```

Part 1. Do a residual analysis by creating a cell below

You should make a plot of the residuals from the fitted model above and explain your findings be the two `---` lines in this cell.

Write your answers here...

In [ ]:
```
### Part 2. Do a Wald Test of H0: beta_1 = 0

# show your working here

XXX
XXX
XXX

# write down your conclusion by replacing XXX in next line
# You will not get points for randomly guessing True/False
RejectNullHypothesisForProblem4 = XXX
```

PROBLEM 4

Maximum Points = 5

1. Take the string `prideAndPrejudiceFirstChapter` and split it by `' '` into a list of "w and put this in `words`.
2. Consider the list of words as a list of states, precisely as in the `wet` - `dry` Markov chain t studied. We model this list of states using a Markov chain, as such there is an associated tra matrix P. The first four words are `['it', 'is', 'a', 'truth']`, if we think of this Markov chain we will have transitions from `'it'` to `'is'` for instance, as such there is a $p_{\textrm{'it','is'}}$, i.e. a transition probability from `'it'` to `'is'`.
3. Your goal is to find the maximum likelihood estimate of P, recall from notebook 13 that for states we have $$ \widehat{p}_{0,0} = \frac{n_{0,0}}{n_{0,0}+n_{0,1}} \quad \text{and} \quad \widehat{p}_{1,1} = \frac{n_{1,1}}{n_{1,0}+n_{1,1}} $$ there is nothing special about two stat for arbitrary number of states $i=1,\ldots,N$ we have $$ \widehat{p}_{i,j} = \frac{n_{i,j}}{\sum ^N n_{i,k}} $$
4. The order of the indices should be the same as the list `unique_words` i.e. the first word i corresponds to $i=0$, the second $i=1$ etc.

In [20]:
```
# REQUIRED-CELL
# DO NOT MODIFY this cell
# Evaluate this cell before trying this PROBLEM so that the required funct
def makeFreqDict(myDataList):
    '''Make a frequency mapping out of a list of data.

    Param myDataList, a list of data.
    Return a dictionary mapping each unique data value to its frequency co

    freqDict = {} # start with an empty dictionary

    for res in myDataList:

        if res in freqDict: # the data value already exists as a key
                freqDict[res] = freqDict[res] + 1 # add 1 to the count usi
        else: # the data value does not exist as a key value
            freqDict[res] = 1 # add a new key-value pair for this new data

    return freqDict # return the dictionary created

# end of makeFreqDict(...)

prideAndPrejudiceFirstChapter = '''It is a truth universally acknowledged
      possession of a good fortune, must be in want of a wife.

      However little known the feelings or views of such a man may be
      on his first entering a neighbourhood, this truth is so well
      fixed in the minds of the surrounding families, that he is
      considered the rightful property of some one or other of their
      daughters.

      “My dear Mr. Bennet,” said his lady to him one day, “have you
      heard that Netherfield Park is let at last?”

      Mr. Bennet replied that he had not.

      “But it is,” returned she; “for Mrs. Long has just been here, and
      she told me all about it.”

      Mr. Bennet made no answer.

      “Do you not want to know who has taken it?” cried his wife
      impatiently.

      “_You_ want to tell me, and I have no objection to hearing it.”

      This was invitation enough.

      “Why, my dear, you must know, Mrs. Long says that Netherfield is
      taken by a young man of large fortune from the north of England;
      that he came down on Monday in a chaise and four to see the
      place, and was so much delighted with it, that he agreed with Mr.
      Morris immediately; that he is to take possession before
      Michaelmas, and some of his servants are to be in the house by
      the end of next week.”

      “What is his name?”

      “Bingley.”
```

In [ ]:
```
# Part 1, find the words by splitting prideAndPrejudiceFirstChapter on '
# Make sure you ran the cell above before you try this
words = XXX
unique_words = sorted(set(words)) # The unique words
n_words = len(unique_words) # The number of unique words
```

In [ ]:
```
# Part 2, count the different transitions
transitions = XXX # A list containing tuples ex: ('it','is') of all transi
transition_counts = XXX # A dictionary that counts the number of each trai
# ex: ('it','is'):4
indexToWord = XXX # A dictionary that maps the n-1 number to the n:th unic
# ex: 0:'a'
wordToIndex = XXX # The inverse function of indexToWord,
# ex: 'a':0
```

In [ ]:
```
# Part 3, finding the maximum likelihood estimate of the transition matri
import numpy as np

transition_matrix = XXX # a numpy array of size (n_words,n_words)

# The transition matrix should be ordered in such a way that
# p_{'it','is'} = transition_matrix[wordToIndex['it'],wordToIndex['is']]

# Make sure that the transition_matrix does not contain np.nan from divis
```

Local Test for PROBLEM 4

Use the cell below to evaluate the feasibility of your answer.

In [ ]:
```
# Once you have created all your functions, you can make a small test here
# what would be generated from your model.

start = np.zeros(shape=(n_words,1))
start[0,0] = 1

current_pos = start
for i in range(100):
    random_word_index = np.random.choice(range(n_words),p=current_pos.resh
    current_pos = np.zeros_like(start)
    current_pos[random_word_index] = 1
    print(indexToWord[random_word_index],end=' ')
    current_pos = (current_pos.T@transition_matrix).T
```

PROBLEM 5

Maximum Points = 5

1. [1p] Draw a uniform random point X on the surface of the unit sphere in $\mathbb{R}^d$. ' the variance of X_1 (the first coordinate)? Solve this using pen and paper, then fill in the a below in `variance_x1_problem7`.
2. [1p] How large must ϵ be for 99% of the volume of a d-dimensional unit-radiu lie in the shell of ϵ-thickness at the surface of the ball?
3. [3p] The volume of the unit ball is given by $$ V(d) = \frac{2 \pi^{\frac{d}{2}}}{d \Gamma(\frac = \frac{ \pi^{\frac{d}{2}}}{ (\frac{d}{2})!} $$ What function of d would the radius need to be f or radius r to have approximately constant volume as a function of d? Hint use Stirlings $n! \approx (n/e)^n$.

In [ ]:
```
# Part1, what is the value of the variance for problem 1
d = var('d')
# Use exact expression, use rationals and not 1.0
variance_x1_problem7 = XXX # Fill this as a function of d (sagemath symbo
```

In [ ]:
```
# Part 2, what is the value of epsilon for question 2
d = var('d')
# Use exact expression, use rationals and not 1.0
epsilon = XXX # Fill this as a function of d (sagemath symbolic expressio
```

In [ ]:
```
# Part 3, what is the radius from problem 3
d = var('d')
# Use exact expression, use rationals and not 1.0.
r = XXX
```

PROBLEM 6

Maximum Points = 5

Consider the data `X` and `y`, in the cell below. `X` denotes 20 points in $\mathbb{R}^2$ and corresponds to the labels for these points, i.e. it is a classification problem.

1. Implement the function `perceptron` by filling in `XXX`.
2. Use your implemented `perceptron` function to compute a vector (numpy array) $\hat w$ shape `(3,1)` such that $$ (\hat w \cdot \hat x_i) l_i > 0, \quad \forall i=1,\ldots,20 $$ put yo answer in `hat_w` below
3. Use the vector $\hat w$ that you just found and compute r (put your result in `r`), finally u to give an upper bound to the number of iterations needed for the perceptron algorithm to co on this dataset, see the Theorem in notebook 15. Put the result in `iteration_bound`.

In [26]:
```
X = np.array([[0.14774693918368506,0.8537253157278155],[-0.17555174302867
y = np.array([1.0,1.0,1.0,1.0,1.0,1.0,1.0,1.0,1.0,1.0,-1.0,-1.0,-1.0,-1.0
```

In [ ]:
```
# Part 1
def perceptron(X_in,labels,max_iter=1000):
    '''Runs the perceptron algorithm on X_in, labels, and does a maximum (
    w = XXX

    return w #Make sure that w has the shape described in the problem

hat_w = XXX
```

In [ ]:
```
# Part 2

r = XXX

iteration_bound = XXX
```

PROBLEM 7

Maximum Points = 5

Perform a bootstrap to find the plug-in estimate and 99% CI for the 95-th Percentile of the inter-E in minutes.

You just need to evaluate the next `REQUIRED-CELL` and replace `XXX` with the right expressio following cell.

NOTE: If `data/earthquakes.csv` is not available and you get a file not found when evaluati next `REQUIRED-CELL`, you can get the csv by `unzip` as follows:

```
%%sh
cd data
unzip earthquakes.csv.zip
```

In [1]:
```
# REQUIRED-CELL
# DO NOT MODIFY this cell
# Evaluate this cell before trying this PROBLEM so that the required funct

import numpy as np
## Be Patient! - This will take more time, about a minute or so
#########################################################################
def getLonLatMagDepTimes(NZEQCsvFileName):
    '''returns longitude, latitude, magnitude, depth and the origin time a
    for each observed earthquake in the csv filr named NZEQCsvFileName'''
    from datetime import datetime
    import time
    from dateutil.parser import parse
    import numpy as np

    with open(NZEQCsvFileName) as f:
        reader = f.read()
        dataList = reader.split('\n')

    myDataAccumulatorList =[]
    for data in dataList[1:-1]:
        dataRow = data.split(',')
        myTimeString = dataRow[2] # origintime
        # let's also grab longitude, latitude, magnitude, depth
        myDataString = [dataRow[4],dataRow[5],dataRow[6],dataRow[7]]
        try:
            myTypedTime = time.mktime(parse(myTimeString).timetuple())
            myFloatData = [float(x) for x in myDataString]
            myFloatData.append(myTypedTime) # append the processed timesta
            myDataAccumulatorList.append(myFloatData)
        except TypeError as e: # error handling for type incompatibilities
            print ('Error:  Error is ', e)
    #return np.array(myDataAccumulatorList)
    return myDataAccumulatorList

myProcessedList = getLonLatMagDepTimes('data/earthquakes.csv')

def interQuakeTimes(quakeTimes):
    '''Return a list inter-earthquake times in seconds from earthquake ori
    Date and time elements are expected to be in the 5th column of the ar
    Return a list of inter-quake times in seconds. NEEDS sorted quakeTimes
    import numpy as np
    retList = []
    if len(quakeTimes) > 1:
        retList = [quakeTimes[i]-quakeTimes[i-1] for i in range(1,len(qual
    #return np.array(retList)
    return retList

def makeBootstrappedConfidenceIntervalOfStatisticT(dataset, statT, alpha,
    '''make a bootstrapped 1-alpha confidence interval for ANY given stat
    from the dataset with B Bootstrap replications for 0 < alpha < 1, and
    return lower CI, upper CI, bootstrapped_samples '''
    n = len(dataset) # sample size of the original dataset
    bootstrappedStatisticTs=[] # list to store the statistic T from each b
    for b in range(B):
        #sample indices at random between 0 and len(iQMinutes)-1 to make
        randIndices=[randint(0,n-1) for i in range(n)]
        bootstrappedDataset = dataset[randIndices] # resample with replace
        bootstrappedStatisticT = statT(bootstrappedDataset)
        bootstrappedStatisticTs.append(bootstrappedStatisticT)
```

In [ ]:
```
# replace XXX with the right expressions
statT95thPercentile = lambda dataset : XXX #statistic of interest (datase
alpha=XXX
B=1000 # number of bootstrap samples, reduce this to 100 while debuging a
# plug-in point estimate of the 75th-Percentile of inter-EQ Times
plugInEstimateOf95thPercentile = XXX
# get the bootstrapped samples and build 1-alpha confidence interval
# do NOT change anything below
lowerCIT95P,upperCIT95P,bootValuesT95P = \
                      makeBootstrappedConfidenceIntervalOfStatisticT(XXX,
print ("The Plug-in Point Estimate of the 95th-Percentile of inter-EQ Time
print ("1-alpha Bootstrapped CI for the 95th-Percentile of inter-EQ Times
print ("         for alpha = ",alpha.n(digits=2)," and bootstrap replicate
```

PROBLEM 8

Maximum Points = 5

Consider n IID samples from a continuous random variable with the following probability densi function: $$ f(x; \beta) = \frac{x}{\beta^2} \exp\left(-\frac{1}{2}(x/\beta)^2\right), \qquad \text{ wher \beta>0, x \geq 0 $$ Use Bounded 1D Optimisation to find the maximum likelihood estimate for th experiment above using the dataset which is given in the numpy array `dataSamplesForProb` below.

In [ ]:
```
import numpy as np
from scipy import optimize

dataSamples1 = np.array([2.30, 4.10, 3.60, 2.50, 3.20, 1.90, 2.60, 1.50, 2

# finding MLE numerically for parameter beta - replace XXX by the right ex
# do NOT change the function name `negLogLklOfIIDSamplesInProblem1or2`
def negLogLklOfIID1(paramBeta):
    '''negative log likelihood function for IID trials in Problem 1 or 2'
    return XXX

# you should NOT change variable names - just replace XXX
boundedResult1 = optimize.minimize_scalar(XXX, XXX, bounds=(XXX, XXX), me
boundedResult1
```