Product Analytics

DIANE WOODBRIDGE, PH.D

Review

Backend and Frontend

Backend and Data Acquisition

- API
- wget
- Crawl
- Collect data from users
  - Create file upload form using Python Flask and WTF.

Contents

Contents

Backend and Frontend

Developing a Login System (Backend)

- User
- Register a user
- Login a user
- Logout a user

Appendix

- Ham and Spam Classification using MongoDB and Spark (Backend).

Full Stack Development

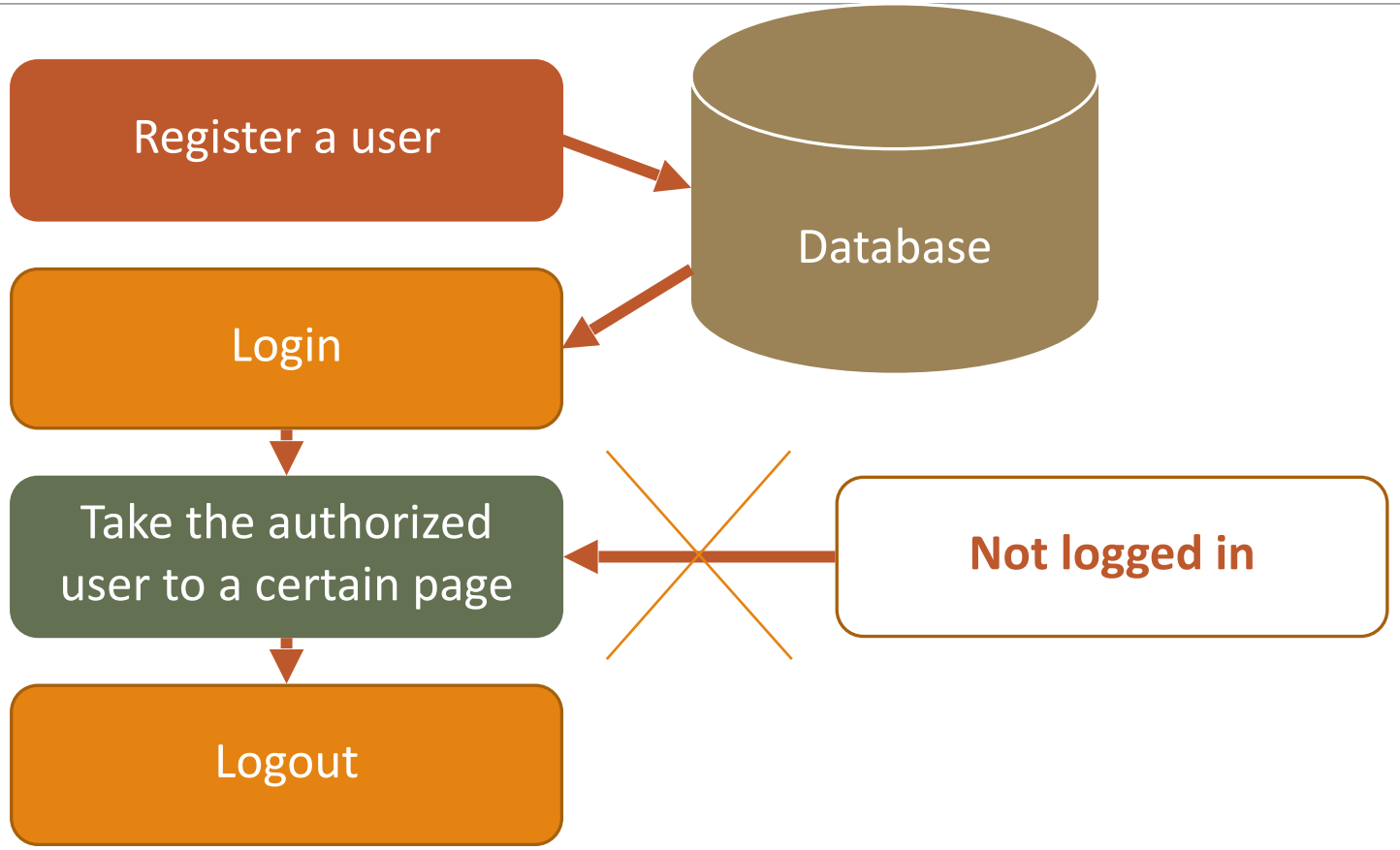
Front end + Back end

Contents

Login System

Take the authorized user to a certain page

Login System



Contents

User

Attributes of a user?

USER

ID

Email

Login password

…

Object Oriented Programming

Class

- Blueprint with 1) attributes and 2) methods for modifying its state.
- Class Definition

```
class ClassName():
    def __init__(self, val): # Initialize the state of an object
        self.attribute = val
    def method(self):
        do/return something
```

- Inheritance : A class can have parent classes and inherit its attributes and methods.
- PEP 8 : Class names should normally use the CapWords convention.

https://docs.python.org/3/tutorial/classes.html

Object Oriented Programming

Object

- Instance of a class that encapsulate class attributes and methods.

```
object = ClassName(val)
```

- When you create an object, **__init__()** is called to initialize the attributes of a class.

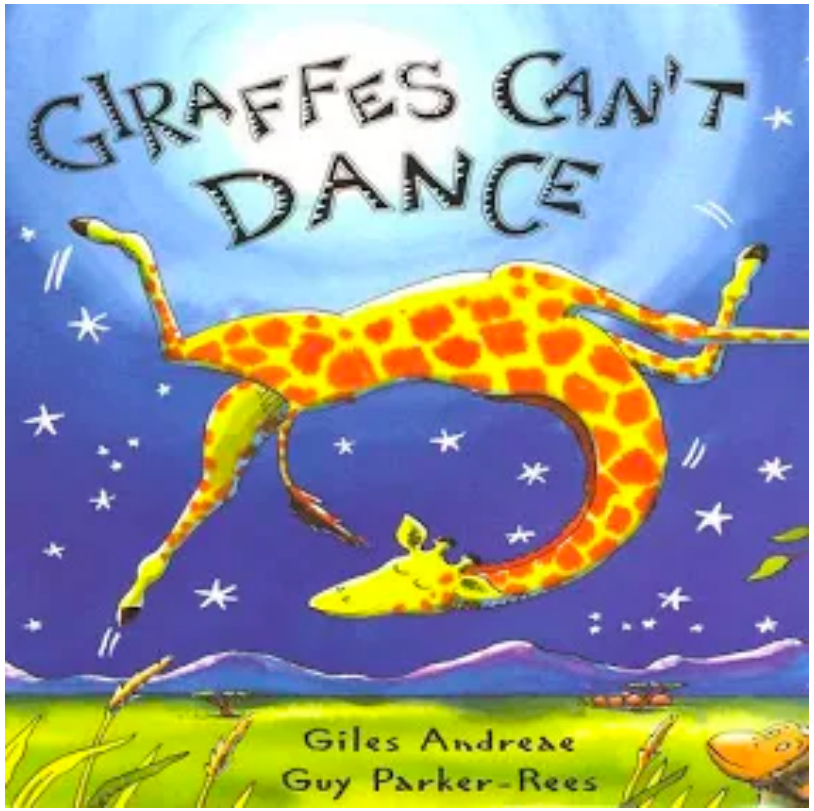
https://docs.python.org/3/tutorial/classes.html

Animals

```
class Animal():
    def __init__(self):
        self.breath = True
        self.eat = True
        self.sleep = True

    def add_favorite_food(self, food):
        self.favorite_food = food
```

oop_example.ipynb

Gerald, the Giraffe




```
gerald = Animal()
gerald.add_favorite_food('leaves')

print(gerald.breath)
print(gerald.eat)
print(gerald.sleep)
print(gerald.favorite_food)
print(gerald.bark_sound())
```

oop_example.ipynb

What is gerald's bark_sound()?

Top

Dogs

```
class Dog(Animal):
    def __init__(self):
        Animal.__init__(self)
        self.bark = True

    def add_favorite_toy(self, toy):
        self.favorite_toy = toy

    def bark_sound(self):
        return ("Arrrrf")


class Newfoundland(Dog):
    def __init__(self):
        Dog.__init__(self)
        self.size = 'Huge'
```

oop_example.ipynb

Boomer, the Newfie

```
boomer = Newfoundland()
boomer.add_favorite_toy('ball')
boomer.add_favorite_food('berries')

print(boomer.breath)
print(boomer.eat)
print(boomer.sleep)
print(boomer.favorite_food)
print(boomer.favorite_toy)
print(boomer.size)
print(boomer.bark_sound())
```

oop_example.ipynb

User

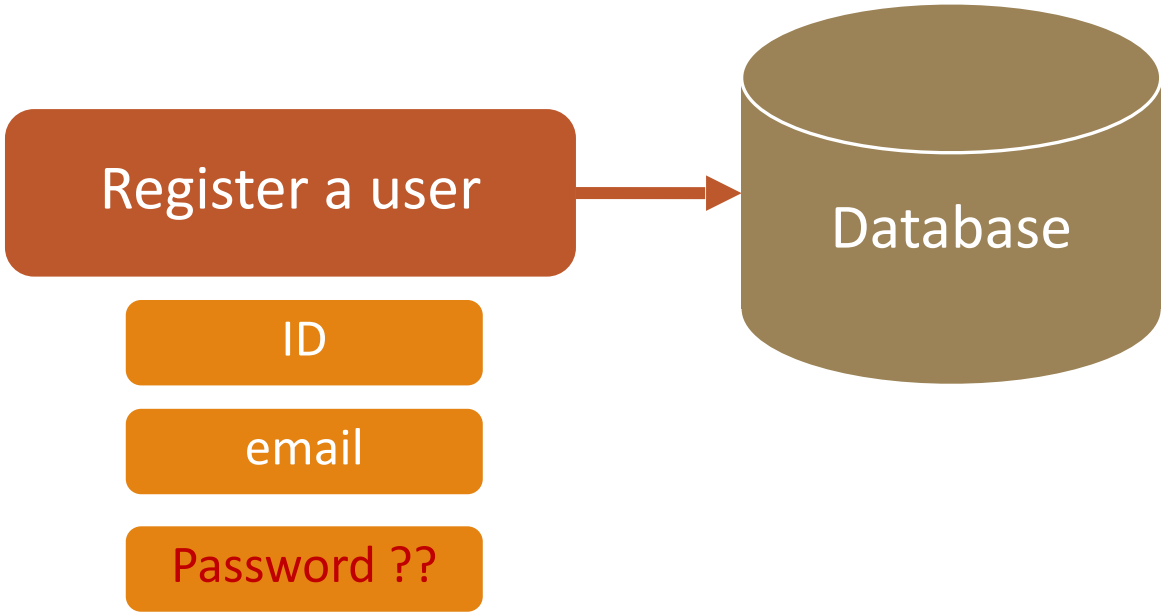
```
class User():
    def __init__(self, username, email, password):
        self.username = username
        self.email = email
        self.password = password


user = User('diane', 'diane@gmail.com', 'pwd')

print(user.username)
print(user.email)
print(user.password)
```

login_step001_user.ipynb

Contents

Register a User - Store User Information

What to store?



Password Security

Can we store the password in a DB?

- Many users use the same password for multiple sites.
  - If your website is hacked, their password can be exposed.

Password Security

Solution?

- Store your password as something else.
  - Password hashing
    - Applies cryptographic transformation to password.
    - It is hard to reverse (decrypt). – For logging in, it performs the same hash function and compares with the stored hashed password.
    - ***werkzeug.security*** provides *generate_password_hash* and *check_password_hash*.
      - ***generate_password_hash****(password, method='pbkdf2:sha256', salt_length=8)*
        - Takes *password* and returns the password hash.
      - ***check_password_hash****(pwhash, password)*
        - Compares *pwhash* with *password* and return True if they are same.

http://werkzeug.pocoo.org/docs/0.14/utils/#module-werkzeug.security

Password Hash

Output : method$salt$hash

```
from werkzeug.security import generate_password_hash
```

generate_password_hash()

creates method$salt$hash format value which is different each time

Same password,
Different hash values.

```
generate_password_hash("1234")
```

```
'pbkdf2:sha256:50000$zIvsp4RJ$03f02285b3a4407fa11788482e5b2fb37bb6e71a47fef76c7cf9bad0c063d6b4'
```

```
generate_password_hash("1234")
```

```
'pbkdf2:sha256:50000$ANeK5ggV$075bac219033a40bb9f6990fc8ec7ff2b5f9e71988035da966f6b6206d0080b9'
```

```
generate_password_hash("1234")
```

```
'pbkdf2:sha256:50000$M8J9tCCf$7d729c34ef3e6088aad4bf51e6fdfa810b6857ef1e555b56a16af9a8a5b91f02'
```

password_hash.ipynb

Password Hash

```
from werkzeug.security import check_password_hash, generate_password_hash


class User():
    def __init__(self, username, email, password):
        self.username = username
        self.email = email
        self.set_password(password)

    def set_password(self, password):
        self.password_hash = generate_password_hash(password)

    def check_password(self, password):
        return check_password_hash(self.password_hash, password)


user = User('diane', 'diane@gmail.com', 'pwd')

print(user.username)
print(user.email)
print(user.password_hash)
print(user.check_password('pwd'))
print(user.check_password('password'))
```

login_step002_password_hash.ipynb

SQLAlchemy

Python SQL toolkit and Object Relational Mapper that gives application developers the full power and flexibility of SQL.

```
$ pip install Flask-SQLAlchemy
```

```
(ProductAnalytics_Env) (base) ML-ITS-901885:2020_MSDS603 dwoodbridge$ pip install Flask-SQLALchemy
Requirement already satisfied: Flask-SQLALchemy in ./ProductAnalytics_Env/lib/python3.7/site-packages (2
Requirement already satisfied: Flask>=0.10 in ./ProductAnalytics_Env/lib/python3.7/site-packages (from F
Requirement already satisfied: SQLAlchemy>=0.8.0 in ./ProductAnalytics_Env/lib/python3.7/site-packages (
Requirement already satisfied: Werkzeug>=0.15 in ./ProductAnalytics_Env/lib/python3.7/site-packages (fro
Requirement already satisfied: click>=5.1 in ./ProductAnalytics_Env/lib/python3.7/site-packages (from Fl
Requirement already satisfied: itsdangerous>=0.24 in ./ProductAnalytics_Env/lib/python3.7/site-packages
Requirement already satisfied: Jinja2>=2.10.1 in ./ProductAnalytics_Env/lib/python3.7/site-packages (fro
```

http://flask-sqlalchemy.pocoo.org/2.3/

SQLAlchemy

Database abstraction layer package that allows you to work at a higher level with regular Python objects instead of database entities.

Ex.

```
cursor = db.cursor()
cursor.execute('SELECT id FROM user WHERE name = name')
user = cursor.fetchone()

vs

user = User.query.filter_by((name = name)).first()
```

http://flask-sqlalchemy.pocoo.org/2.3/

SQLAlchemy

Benefits

- Ease of use : Transparent conversion of high-level object-oriented operations into low-level db queries.
  - However, conversions from objects to relationships can degrade performance.
- Portability : With the same code, you can change your database types (Postgres, MySQL, SQLite, etc.).

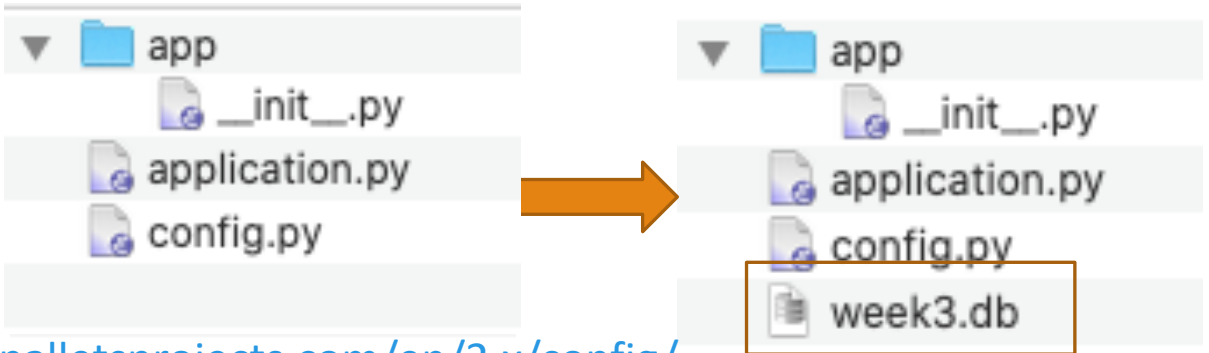
Database Engine	URI
Postgres	postgresql://username:password@hostname/database
MySQL	mysql://username:password@hostname/database
SQLite	sqlite:////absolute_path_to_database sqlite:///database_in_current_directory
Oracle	oracle://username:password@@127.0.0.1:1521/sidname

http://flask-sqlalchemy.pocoo.org/2.3/

SQLAlchemy : login_step003_SQLAlchemy_config

```
import os
basedir = os.path.abspath(os.path.dirname(__file__))


class Config(object):
    SQLALCHEMY_DATABASE_URI = 'sqlite:///' + os.path.join(basedir, 'week3.db')
    SQLALCHEMY_TRACK_MODIFICATIONS = True
```

config.py



https://flask-sqlalchemy.palletsprojects.com/en/2.x/config/

SQLAlchemy : login_step003_SQLAlchemy_config

```
from flask import Flask
from flask_sqlalchemy import SQLAlchemy

from config import Config


# Initialization
# Create an application instance (an object of class Flask)  which handles all requests.
application = Flask(__name__)
application.config.from_object(Config)

db = SQLAlchemy(application)
db.create_all()
db.session.commit()
```

create all the tables and databases.

finalize changes.

__init__.py

SQLAlchemy

Database Model Definition

- A Python class with attributes that match the columns of a corresponding database.
- Pass your *db.Model* to the class which is a baseclass for your model.
- Define attributes of the model using *db.column(column_type, column_options).*

Column Type Name	Description
Integer	an integer
String(size)	a string with a maximum length (optional in some databases, e.g. PostgreSQL)
Text	some longer unicode text
DateTime	date and time expressed as Python datetime object.
Float	stores floating point values
Boolean	stores a boolean value
PickleType	stores a pickled Python object
LargeBinary	stores large arbitrary binary data

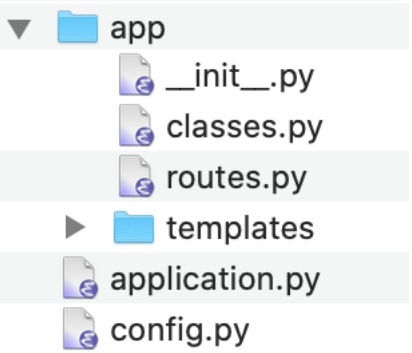
SQLAlchemy

Database Model Definition

- A Python class with attributes that match the columns of a corresponding database.
- Pass your *db.Model* to the class which is a baseclass for your model.
- Define attributes of the model using *db.column(column_type, column_options).*

Column Option Name	Description
primary_key	If set to True, the column is the table's primary key.
unique	If set to True, do not allow duplicate values for this column.
index	If set to True, create an index for this column for efficient query performance.
nullable	If set to True, allow empty values.
default	Assign a default value for this column.

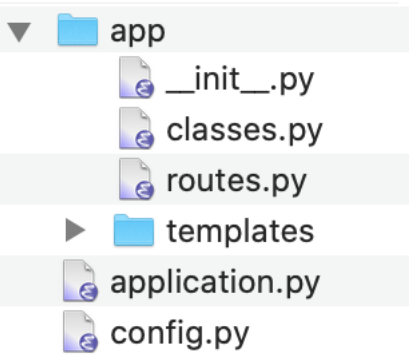
Store User into a DB : login_step004_SQLAlchemy



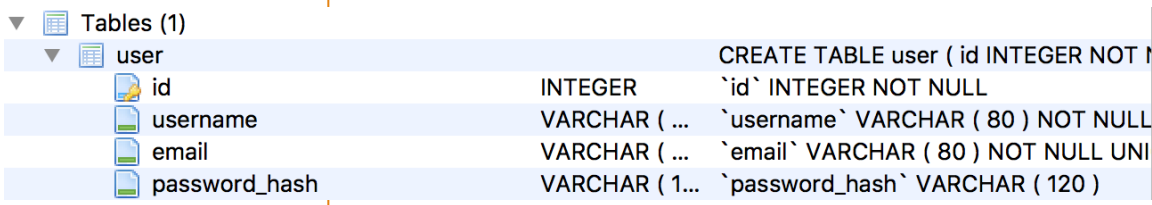

```python
from flask import Flask
from config import Config
from flask_sqlalchemy import SQLAlchemy

# Initialization
# Create an application instance (an object of class Flask)  which handles all requests.
application = Flask(__name__)
application.config.from_object(Config)
db = SQLAlchemy(application)
db.create_all()
db.session.commit()

# Added at the bottom to avoid circular dependencies. (Altough it violates PEP8 standards)
from app import classes
from app import routes
```

__init__.py

Store User into a DB : login_step004_SQLAlchemy

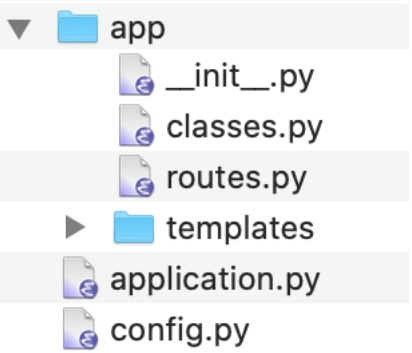



```
from werkzeug.security import check_password_hash
from werkzeug.security import generate_password_hash

from app import db

class User(db.Model):
    id = db.Column(db.Integer, primary_key=True)
    username = db.Column(db.String(80), unique=True, nullable=False)
    email = db.Column(db.String(80), unique=True, nullable=False)
    password_hash = db.Column(db.String(120), nullable=False)

    def __init__(self, username, email, password):
        self.username = username
        self.email = email
        self.set_password(password)

    def set_password(self, password):
        self.password_hash = generate_password_hash(password)

    def check_password(self, password):
        return check_password_hash(self.password_hash, password)

db.create_all()
db.session.commit()
```



classes.py

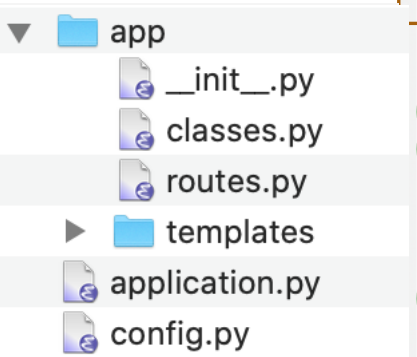
Store User into a DB : login_step004_SQLAlchemy




```python
class RegistrationForm(FlaskForm):
    username = StringField('Username:', validators=[DataRequired()])
    email = StringField('Email:', validators=[DataRequired()])
    password = PasswordField('Password:', validators=[DataRequired()])
    submit = SubmitField('Submit')
```

classes.py

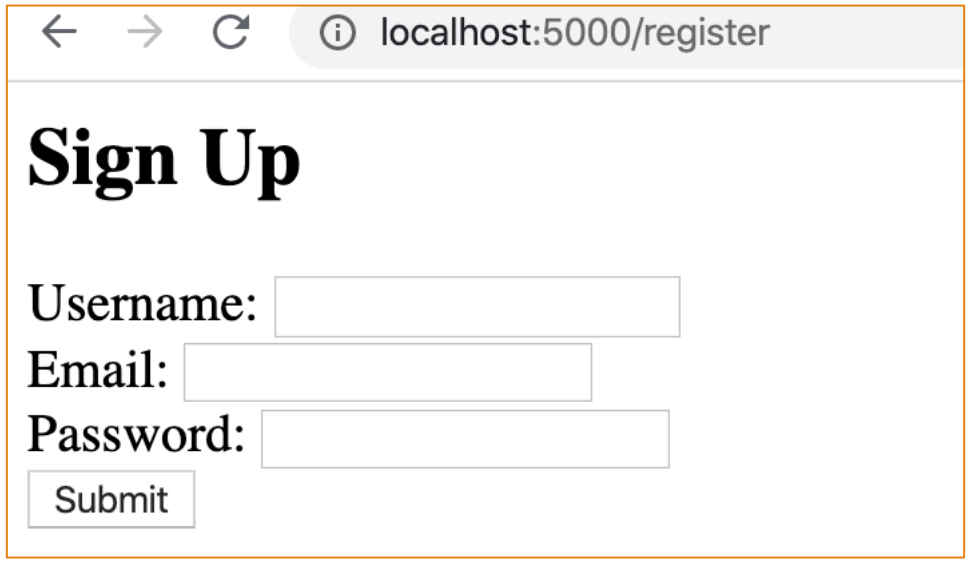
Store User into a DB : login_step004_SQLAlchemy

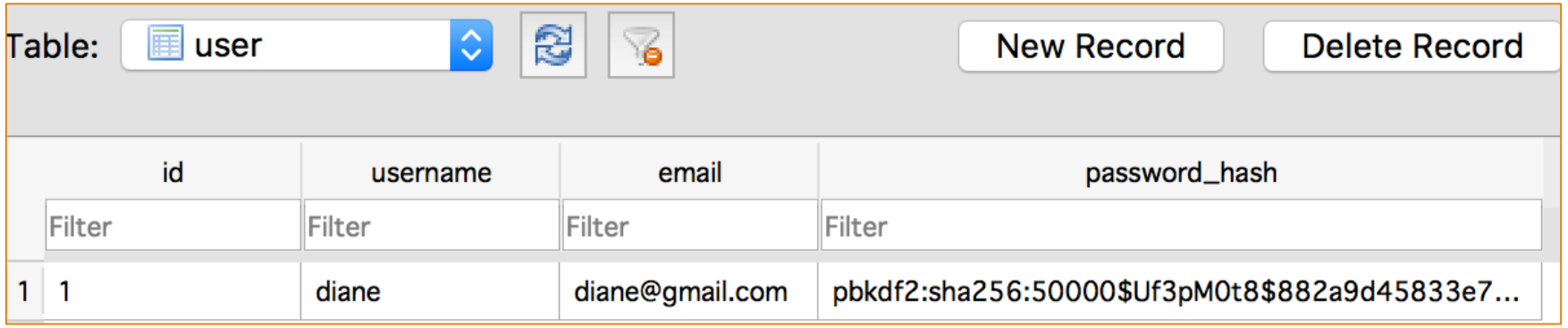



```python
from app import application, classes, db
from flask import render_template, redirect, url_for

@application.route('/index')
@application.route('/')
def index():
    return("<h1> Hello World </h1>")

@application.route('/register',  methods=('GET', 'POST'))
def register():
    registration_form = classes.RegistrationForm()
    if registration_form.validate_on_submit():
        username = registration_form.username.data
        password = registration_form.password.data
        email = registration_form.email.data
        ##################################
        #### UPDATE THIS (EXERCISE 1) ####
        ##################################
        user = classes.User(username, email, password)
        db.session.add(user)
        db.session.commit()
        return redirect(url_for('index'))
    return render_template('register.html', form=registration_form)
```

routes.py

localhost/register





Exercise 1

If a username is already registered, throw an error.

```
  File "/Users/dwoodbridge/anaconda3/envs/MSDS603/lib/python3.7/site-packages/sqlalchemy/engine/default.py", line 552, in do_execute
    cursor.execute(statement, parameters)
sqlalchemy.exc.IntegrityError: (sqlite3.IntegrityError) UNIQUE constraint failed: user.email
[SQL: INSERT INTO user (username, email, password_hash) VALUES (?, ?, ?)]
[parameters: ('diane', 'diane@gmail.com', 'pbkdf2:sha256:50000$y4CZEH7e$f35a6bd9adffb51660f4e71e4f7eeb7e735da82d6fe9867ee9c9b9a203625e5
1')]
(Background on this error at: http://sqlalche.me/e/gkpj)
```

Contents

flask-login

Benefits

- Store the active user's ID in the session, and let you log them in and out.
- You can restrict views to logged-in (or logged-out) users.
- Handle the normally-tricky "remember me" functionality.

Installation

```
$ pip install flask-login
```

```
(ProductAnalytics_Env) (base) ML-ITS-901885:2020_MSDS603 dwoodbridge$ pip install flask-login
Requirement already satisfied: flask-login in ./ProductAnalytics_Env/lib/python3.7/site-packages (0.5.0)
Requirement already satisfied: Flask in ./ProductAnalytics_Env/lib/python3.7/site-packages (from flask-login) (1.1.1)
Requirement already satisfied: click>=5.1 in ./ProductAnalytics_Env/lib/python3.7/site-packages (from Flask->flask-login) (7.1.1)
Requirement already satisfied: Werkzeug>=0.15 in ./ProductAnalytics_Env/lib/python3.7/site-packages (from Flask->flask-login) (1.0.0)
Requirement already satisfied: Jinja2>=2.10.1 in ./ProductAnalytics_Env/lib/python3.7/site-packages (from Flask->flask-login) (2.11.1)
Requirement already satisfied: itsdangerous>=0.24 in ./ProductAnalytics_Env/lib/python3.7/site-packages (from Flask->flask-login) (1.1.0)
Requirement already satisfied: MarkupSafe>=0.23 in ./ProductAnalytics_Env/lib/python3.7/site-packages (from Jinja2>=2.10.1->Flask->flask-login) (1.1.1)
```

https://flask-login.readthedocs.io

flask-login : login_step005_login

LoginManager

- The LoginManager class lets your application and flask-login work together.
  - How to load a user from an ID, Where to send users, When they need to log in, etc.
- Once the actual application object has been created, you can configure it for login by using *init_app*.

__init__.py

```
from flask import Flask
from config import Config
from flask_sqlalchemy import SQLAlchemy
from flask_login import LoginManager

# Initialization
# Create an application instance (an object of class Flask)  which handles all requests.
application = Flask(__name__)
application.config.from_object(Config)
db = SQLAlchemy(application)
db.create_all()
db.session.commit()

# login_manager needs to be initiated before running the app
login_manager = LoginManager()
login_manager.init_app(application)

# Added at the bottom to avoid circular dependencies. (Altough it violates PEP8 standards)
from app import classes
from app import routes
```

flask-login : login_step005_login

```
@application.route('/login', methods=['GET', 'POST'])
def login():
    username = 'diane'
    password = 'pwd'
    # Look for it in the database.
    user = classes.User.query.filter_by(username=username).first()

    # Login and validate the user.
    if user is not None and user.check_password(password):
        login_user(user)
        return '<h1> Logged in : ' + username + '</h1>'
    else:
        return '<h1> Invalid username and password combination! </h1>'
```

routes.py

flask-login : login_step005_login

user_loader

- Reload a user from the session.
- Take a user ID and return a user object, or None if the user does not exist.

```
@login_manager.user_loader
def load_user(id):  # id is the ID in User.
    return User.query.get(int(id))
```

classes.py

localhost/login

```
Traceback (most recent call last):
  File "/Users/dwoodbridge/anaconda3/envs/MSDS603/lib/python3.7/site-packages/flask/app.py", line 2292, in wsgi_app
    response = self.full_dispatch_request()
  File "/Users/dwoodbridge/anaconda3/envs/MSDS603/lib/python3.7/site-packages/flask/app.py", line 1815, in full_dispatch_request
    rv = self.handle_user_exception(e)
  File "/Users/dwoodbridge/anaconda3/envs/MSDS603/lib/python3.7/site-packages/flask/app.py", line 1718, in handle_user_exception
    reraise(exc_type, exc_value, tb)
  File "/Users/dwoodbridge/anaconda3/envs/MSDS603/lib/python3.7/site-packages/flask/_compat.py", line 35, in reraise
    raise value
  File "/Users/dwoodbridge/anaconda3/envs/MSDS603/lib/python3.7/site-packages/flask/app.py", line 1813, in full_dispatch_request
    rv = self.dispatch_request()
  File "/Users/dwoodbridge/anaconda3/envs/MSDS603/lib/python3.7/site-packages/flask/app.py", line 1799, in dispatch_request
    return self.view_functions[rule.endpoint](**req.view_args)
  File "/Users/dwoodbridge/Class/2019_MSDS603/Examples/Week3/Flask_Login_Backend/login_step005_login/code/app/routes.py", line 31, in l
ogin
    login_user(user)
  File "/Users/dwoodbridge/anaconda3/envs/MSDS603/lib/python3.7/site-packages/flask_login/utils.py", line 158, in login_user
    if not force and not user.is_active:
AttributeError: 'User' object has no attribute 'is_active'
```

flask-login : login_step005_login

To use flask-login, you need to **implement** the following methods.

- is_authenticated() : Check whether the user provided valid credentials.
- is_active() : Check whether the user is an active user – an authenticated user that has a non-suspended account.
- is_anonymous() : Check whether the user is anonymous.
- get_id() : Return a unicode that uniquely identifies the user.

➔ Instead, you can inherit from **UserMixin**, which provides default implementations.

Exercise 2

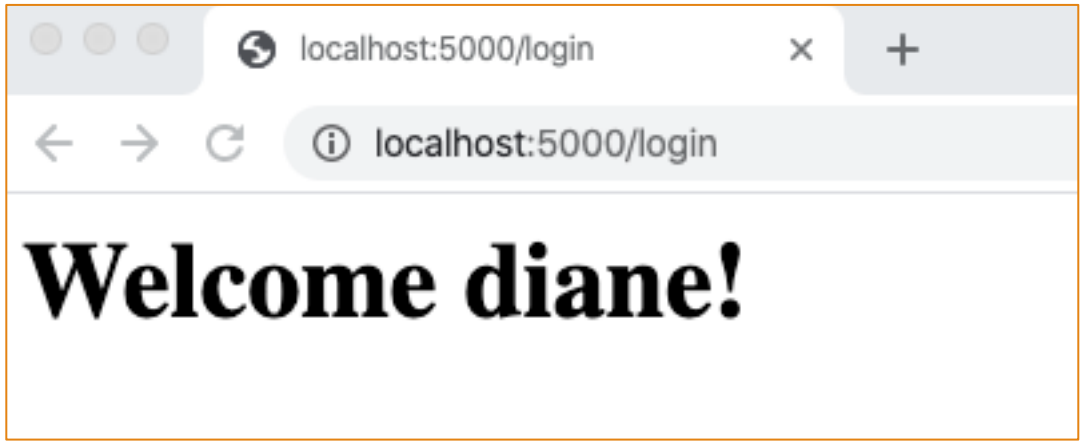
flask-login's ***UserMixin*** provides default implementations for the methods that flask-login expects user objects to have.

```
class User(db.Model, UserMixin):
    id = db.Column(db.Integer, primary_key=True)
    username = db.Column(db.String(80), unique=True, nullable=False)
    email = db.Column(db.String(80), unique=True, nullable=False)
    password_hash = db.Column(db.String(120))

    def __init__(self, username, email, password):
        self.username = username
        self.email = email
        self.set_password(password)

    def set_password(self, password):
        self.password_hash = generate_password_hash(password)

    def check_password(self, password):
        return check_password_hash(self.password_hash, password)
```

classes.py

https://flask-login.readthedocs.io/en/latest/#flask_login.UserMixin

localhost/login



Exercise 3

When a user logged in, redirect the user to *secret_page*.

flask-login : login_step005_login

Flask-login provides *@login_required* decorator.

- If the route is accessed by a non-authenticated user, it will intercept the request.

https://flask-login.readthedocs.io/en/latest/#flask_login.login_required

Exercise 3

Make sure that *secret_page* can be only accessed by authenticated users.

Contents

Backend and Frontend

Developing a Login System (Backend)

- User
- Register a user
- Login a user
- **Logout a user**

Appendix

- Ham and Spam Classification using MongoDB and Spark (Backend).

flask_login : login_step006_logout

Logout

- *logout_user()* removes and resets the user session.

```
@application.route('/logout')
@login_required
def logout():
    before_logout = '<h1> Before logout - is_autheticated : ' \
                    + str(current_user.is_authenticated) + '</h1>'

    logout_user()

    after_logout = '<h1> After logout - is_autheticated : ' \
                   + str(current_user.is_authenticated) + '</h1>'
    return before_logout + after_logout
```

routes.py

localhost/logout

Before logout - is_autheticated : True

After logout - is_autheticated : False

Call it again.

Before logout - is_autheticated : False

After logout - is_autheticated : False

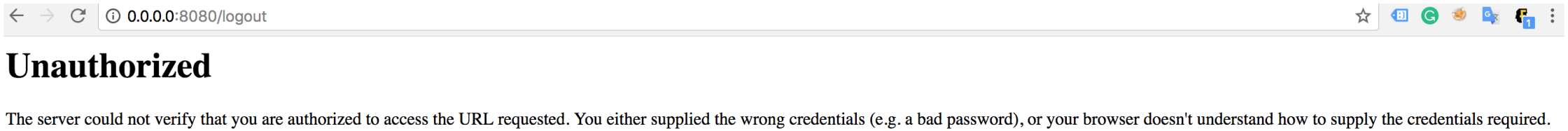
flask_login : login_step006_logout

Add @login_required

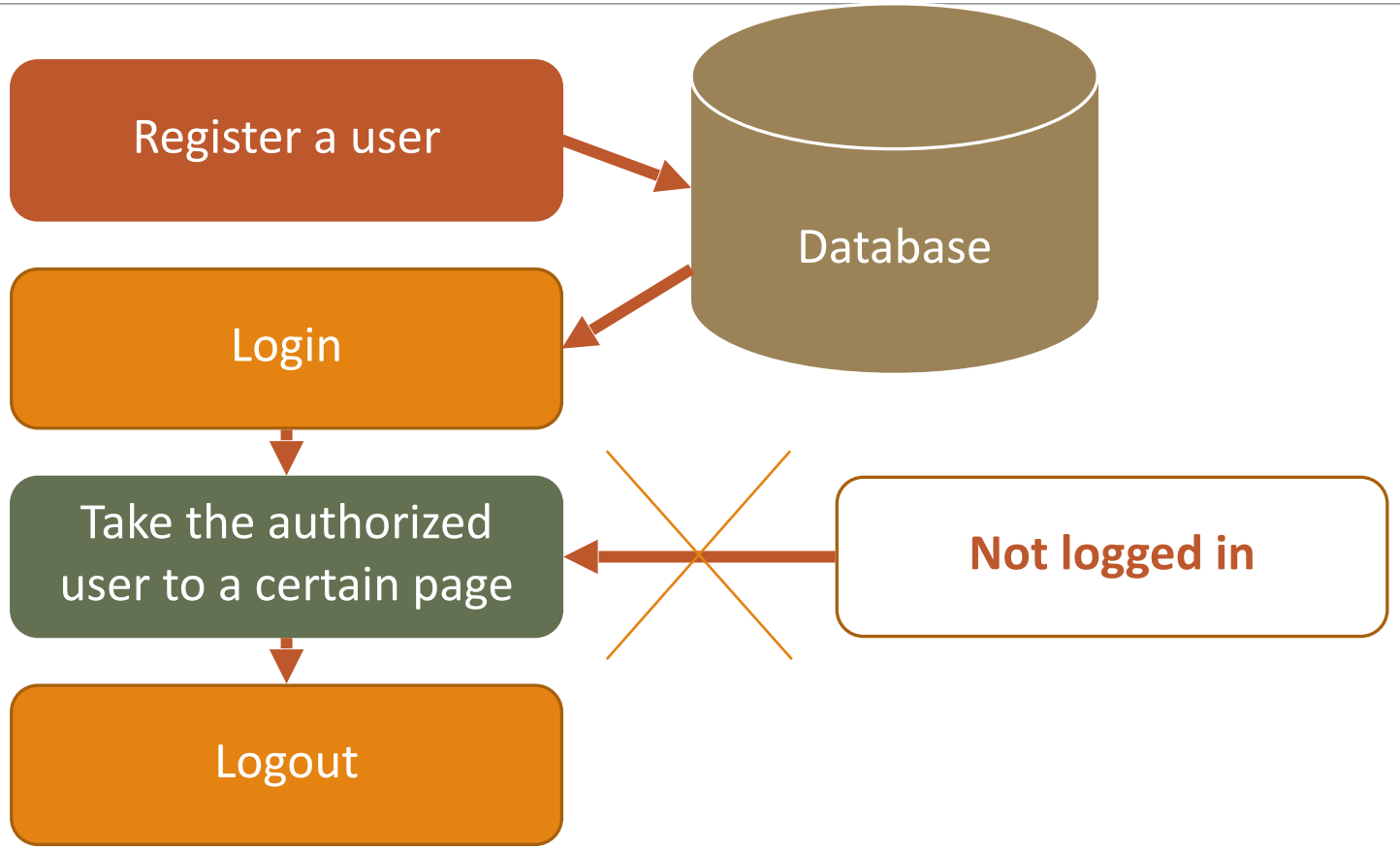
routes.py

```
@app.route('/logout')
@login_required
def logout():
    before_logout = '<h1> Before logout - is_autheticated : ' \
                    + str(current_user.is_authenticated) + '</h1>'

    logout_user()

    after_logout = '<h1> After logout - is_autheticated : ' \
                   + str(current_user.is_authenticated) + '</h1>'
    return before_logout + after_logout
```

If you call localhost/logout twice,



Login System



Contents

Announcement

Who Should I talk to?

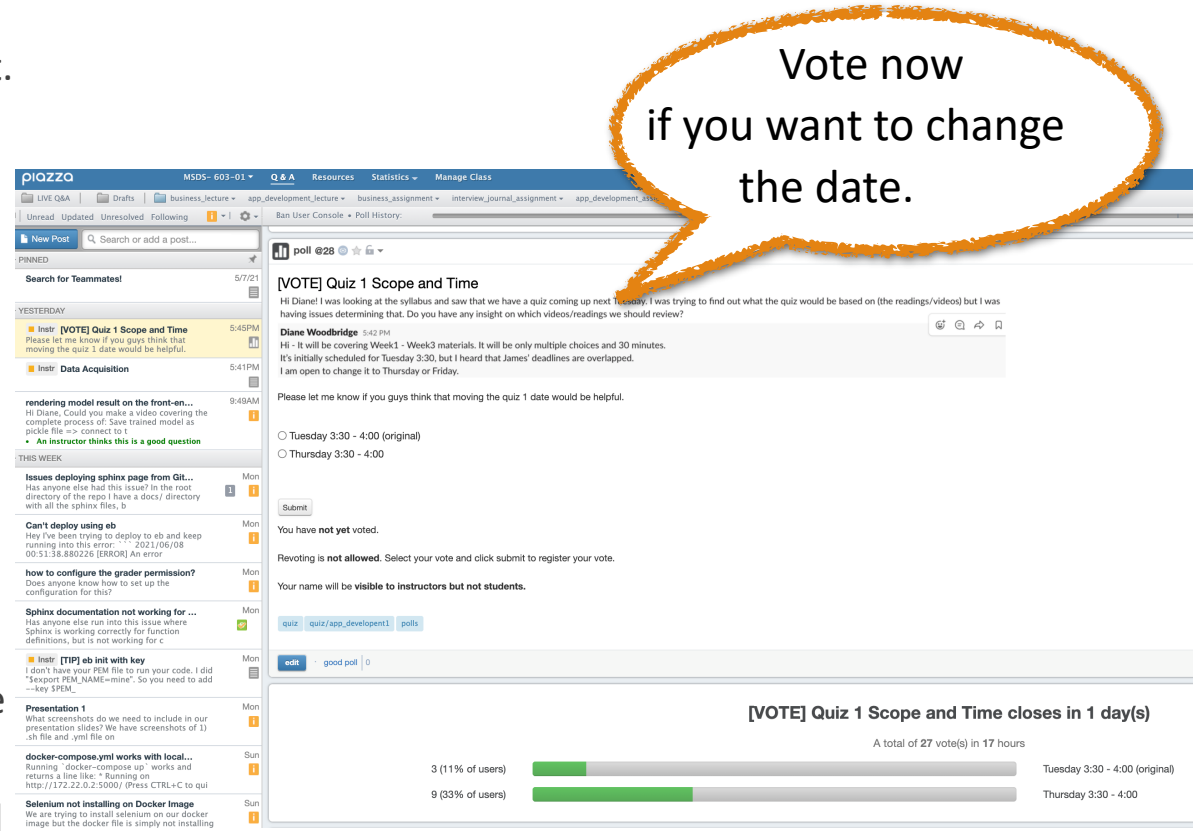
- Business Class/Assignment/Quiz ==> Schaffer
- Application Class/Assignment/Quiz ==> Diane
  - If you get stuck for more than 1 hour on one thing, post on Piazza.

Table 1: Grading Breakdown

Date	Type	Name	Details	Pct	Instructor
5/27	Group	Team and business topic proposal		4	S & D
6/2	Group	Assign 1	Assignment 1: Interviewing plan for team	5	S
6/7	Group	Interview Journal 1		2	S
6/7	Group	Application Phase 1	Presentation, Deploy App, Sphinx, CTest, Trello	5	D
6/7	Individual	Group Peer Review 1		1	D
6/7	Individual	Scoresheet 1		1	S
6/10	Individual	Presentation Review	Grp 1 Only	1	D
6/11	Group	Assign 2	Assignment 2: Industry Analysis and Porters Forces	5	S
6/14	Group	Interview Journal 2		2	S
6/14	Group	Application Phase 2	Presentation, Initial Back-end (Data storage and processing), Update Deploy, Sphinx, CTest, Trello	6	D
6/14	Individual	Group Peer Review 2		1	D
6/14	Individual	Scoresheet 2		1	S
6/15	Individual	Quiz 1		7	D
6/17	Individual	Presentation Review	Grp 2 Only	1	D
6/18	Group	Initial Final Pitch Deck		5	S
6/21	Group	Interview Journal 3		2	S
6/21	Individual	Scoresheet 3		1	S
6/24	Group	Application Phase 3	Presentation, Final Product, Update Deploy, Sphinx, CTest, Trello	8	D
6/24	Individual	Group Peer Review 3		1	D
6/25	Individual	Quiz 2		7	D
6/27	Individual	Final Presentation Review	Everyone	1	D
6/29	Group	VC Presentation Final Pitch Deck		10	S & D
7/1	Individual	Entrepreneur Final Exam		12	S
	Individual	Professionalism		12	S & D

Announcement

1. If you're in Group Presentation Peer Review Date 1, finish watching the video (MUST) and submit your reviews by tonight.
2. App Update 2 is due by Monday midnight.
3. Quiz 1 is scheduled on Tuesday.
    - Week1 - Week3 materials
    - Only multiple choices (30 min)
    - Study list is available.
4. Several optional videos are available.
    - S3, RDS with Flask
    - ML model with Flask
5. Make sure to update Trello regularly - doing it at the last minute will be reflected to the grade.
6. If you are coming in-person office hour, please bring masks, and do Dons health check. And meet me at 9 AM at the front door.



Questions

Flask Application Package Organization

Single module

- It can get messy!
  - Functions, classes, routes, etc. all together?!🤮

```
app.py
config.py
requirements.txt
static/
templates/
```

Multiple module

- Group the different components in a logical way.

```
config.py
requirements.txt
run.py
instance/
    config.py
yourapp/
    __init__.py
    views.py
    models.py
    forms.py
    static/
    templates/
```

https://exploreflask.com/en/latest/organizing.html

Flask Application Package Organization

Multiple module

run.py (application.py in our codebase)	This is the file that is invoked to start up a development server. It gets a copy of the app from your package and runs it.
requirements.txt	This file lists all of the Python packages that your app depends on.
config.py	This file contains most of the configuration variables that your app needs.
/yourapp/	This is the package that contains your application.
/yourapp/__init__.py	This file initializes your application and brings together all of the various components.
/yourapp/views.py (routes.py in our coudebase)	This is where the routes are defined.
/yourapp/models.py (classes.py)	This is where you define the models of your application. This may be split into several modules in the same way as views.py.
/yourapp/static/	This directory contains the public CSS, JavaScript, images and other files that you want to make public via your app. It is accessible from yourapp.com/static/ by default.
/yourapp/templates/	This is where you'll put the Jinja2 templates for your app.

```
config.py
requirements.txt
run.py
instance/
    config.py
yourapp/
    __init__.py
    views.py
    models.py
    forms.py
    static/
    templates/
```

https://exploreflask.com/en/latest/organizing.html

Unit Test in Extended Codebase.

Need a frequent update in your test_*.py (*_test.py)

```python
from app import classes
from app import db
from config import Config
from flask_sqlalchemy import SQLAlchemy

def UserFromDB(username):
    user = classes.User.query.filter_by(username=username).first()
    return user

def test_db_existence():
    """
    Check whether the __init__ created a db and user class table.
    """
    db = SQLAlchemy()
    engine = db.create_engine(Config.SQLALCHEMY_DATABASE_URI, {})
    inspect = db.inspect(engine)
    assert (inspect.has_table("user"))

def test_UserFromDB():
    """
    Assuming that "DIANE, diane@gmail.com, 1234" is always in the database
    Good to have a test user account
    """
    assert UserFromDB("DIANE").email == "d@gmail.com"
    assert UserFromDB("DIANE").username == "DIANE"
```

Week3/RDS_Example/login_step007_AWS_RDS/test_app.py

Contents

Questions

Appendix

HAM AND SPAM CLASSIFICATION USING MONGODB AND SPARK.

Spark MLlib

Algorithms

- Basic Statistics : summary statistics, correlations, stratified sampling, hypothesis testing, streaming significance testing, random data generation
- Classification and Regression : Linear Support Vector Machines (SVMs), Logistic regression, Linear least squares, Lasso, and ridge regression, naive Bayes decision trees ensembles of trees (Random Forests and Gradient-Boosted Trees) isotonic regression
- Collaborative Filtering : alternating least squares (ALS)
- Clustering : k-means, Gaussian mixture, power iteration clustering (PIC), latent Dirichlet allocation (LDA), bisecting k-means, streaming k-means

https://spark.apache.org/docs/2.0.0-preview/mllib-guide.html

Spark MLlib

Algorithms

- Dimensionality Reduction : singular value decomposition (SVD), principal component analysis (PCA)
- Feature extraction and transformation
- Frequent Pattern Mining : FP-growth, association rules, PrefixSpan

https://spark.apache.org/docs/2.0.0-preview/mllib-guide.html

Spark MLlib

Logistic Regression

- Use set of independent variables to make predictions about a target variable (label).
- Assumption : a linear relationship between the independent and target variables.

- `LogisticRegressionWithSGD`
  - Logistic regression using stochastic gradient descent.

Spark MLlib – spark_pam_spam_classification

Logistic Regression Example

- Dataset
  - Spam
  - Ham

Training Data

Data

- SMS Spam Collection Data Set, Machine Learning Repository, University of California, Irvine.
  - Source : https://archive.ics.uci.edu/ml/datasets/sms+spam+collection
  - Size : 5574 SMS messages (747 spam and 4827 ham messages.)
  - Format : TSV

```
ham  Go until jurong point, crazy.. Available only in bugis n great world la e buffet... Cine there got
ham  Ok lar... Joking wif u oni...
spam Free entry in 2 a wkly comp to win FA Cup final tkts 21st May 2005. Text FA to 87121 to receive ent
ham  U dun say so early hor... U c already then say...
ham  Nah I don't think he goes to usf, he lives around here though
spam FreeMsg Hey there darling it's been 3 week's now and no word back! I'd like some fun you up for it
ham  Even my brother is not like to speak with me. They treat me like aids patent.
ham  As per your request 'Melle Melle (Oru Minnaminunginte Nurungu Vettam)' has been set as your callert
spam WINNER!! As a valued network customer you have been selected to receivea £900 prize reward! To clai
spam Had your mobile 11 months or more? U R entitled to Update to the latest colour mobiles with camera
ham  I'm gonna be home soon and i don't want to talk about this stuff anymore tonight, k? I've cried eno
spam SIX chances to win CASH! From 100 to 20,000 pounds txt> CSH11 and send to 87575. Cost 150p/day, 6da
spam URGENT! You have won a 1 week FREE membership in our £100,000 Prize Jackpot! Txt the word: CLAIM to
ham  I've been searching for the right words to thank you for this breather. I promise i wont take your
ham  I HAVE A DATE ON SUNDAY WITH WILL!!
spam XXXMobileMovieClub: To use your credit, click the WAP link in the next txt message or click here>>
ham  Oh k...i'm watching here:)
ham  Eh u remember how 2 spell his name... Yes i did. He v naughty make until i v wet.
ham  Fine if that□s the way u feel. That□s the way its gota b
spam England v Macedonia - dont miss the goals/team news. Txt ur national team to 87077 eg ENGLAND to 87
ham  Is that seriously how you spell his name?
```

Testing Data

Data

- Assumption
  - A text message response is sent to a server using services like twilio.
  - Format : Free text.

Data

Database : MongoDB

- Collection : msan698
- Database : sms

```
> db.sms.find()
{ "_id" : ObjectId("5aa9896e5c50bf0be613d330"), "Label" : "ham", "Message" : "Go until jurong point, crazy.. Available only in bugis n g
reat world la e buffet... Cine there got amore wat..." }
{ "_id" : ObjectId("5aa9896e5c50bf0be613d331"), "Label" : "spam", "Message" : "WINNER!! As a valued network customer you have been selec
ted to receivea £900 prize reward! To claim call 09061701461. Claim code KL341. Valid 12 hours only." }
{ "_id" : ObjectId("5aa9896e5c50bf0be613d332"), "Label" : "spam", "Message" : "Had your mobile 11 months or more? U R entitled to Update
 to the latest colour mobiles with camera for Free! Call The Mobile Update Co FREE on 08002986030" }
{ "_id" : ObjectId("5aa9896e5c50bf0be613d333"), "Label" : "ham", "Message" : "I'm gonna be home soon and i don't want to talk about this
 stuff anymore tonight, k? I've cried enough today." }
{ "_id" : ObjectId("5aa9896e5c50bf0be613d334"), "Label" : "ham", "Message" : "As per your request 'Melle Melle (Oru Minnaminunginte Nuru
ngu Vettam)' has been set as your callertune for all Callers. Press *9 to copy your friends Callertune" }
{ "_id" : ObjectId("5aa9896e5c50bf0be613d335"), "Label" : "spam", "Message" : "SIX chances to win CASH! From 100 to 20,000 pounds txt> C
SH11 and send to 87575. Cost 150p/day, 6days, 16+ TsandCs apply Reply HL 4 info" }
{ "_id" : ObjectId("5aa9896e5c50bf0be613d336"), "Label" : "ham", "Message" : "I've been searching for the right words to thank you for t
his breather. I promise i wont take your help for granted and will fulfil my promise. You have been wonderful and a blessing at all time
s." }
{ "_id" : ObjectId("5aa9896e5c50bf0be613d337"), "Label" : "ham", "Message" : "I HAVE A DATE ON SUNDAY WITH WILL!!" }
{ "_id" : ObjectId("5aa9896e5c50bf0be613d338"), "Label" : "spam", "Message" : "XXXMobileMovieClub: To use your credit, click the WAP lin
k in the next txt message or click here>> http://wap. xxxmobilemovieclub.com?n=QJKGIGHJJGCBL" }
{ "_id" : ObjectId("5aa9896e5c50bf0be613d339"), "Label" : "ham", "Message" : "Oh k...i'm watching here:)" }
{ "_id" : ObjectId("5aa9896e5c50bf0be613d33a"), "Label" : "spam", "Message" : "URGENT! You have won a 1 week FREE membership in our £100
,000 Prize Jackpot! Txt the word: CLAIM to No: 81010 T&C www.dbuk.net LCCLTD POBOX 4403LDNW1A7RW18" }
{ "_id" : ObjectId("5aa9896e5c50bf0be613d33b"), "Label" : "ham", "Message" : "U dun say so early hor... U c already then say..." }
{ "_id" : ObjectId("5aa9896e5c50bf0be613d33c"), "Label" : "ham", "Message" : "Fine if thats the way u feel. Thats the way its gota b"
```

Insert Training Data

```
def insert_data_file(input, database, collection, format, fields):
    """
    Read input data as TSV and insert into the collection.
    """

    mongoimport_query =  "mongoimport" +  " --db " + database + " --collection " \
                        + collection + " --file " + input + " --type " \
                        + format + " --fields " + fields

    subprocess.call(mongoimport_query, shell=True)
```

Output :

```
> db.sms.count()
5574
```

Insert Data with a Label

```
def insert_data_entry(input, fields, delimiter, database, collection):
    """
    Read a single line input and insert into collection.
    """
    db_conn = create_connection(database)

    splitted_fields = fields.split(delimiter)
    splitted_input  = input.split(delimiter)

    data = {}
    for count in range(0, fields.count(delimiter)+1):
        data[splitted_fields[count]] = splitted_input[count]
        json_data = json.dumps(data)

    db_conn[collection].insert_one(data)
```

Output :

```
> db.sms.count()
5575
```

```
> db.sms.find()
{ "_id" : ObjectId("5aa9a40a58a8860bc68b4722"), "Label" : "ham", "Message" : "How are you?" }
```

Load Data from MongoDB to Spark

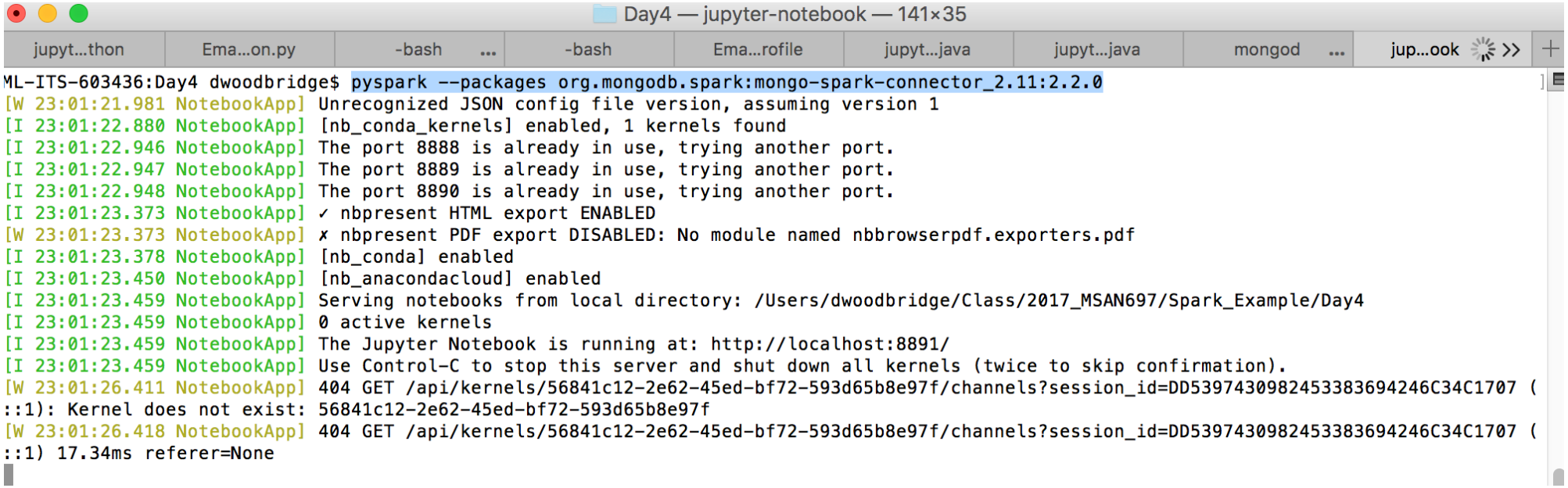
Start your mongod process

```
ML-ITS-603436:Answer dwoodbridge$ mongod
2018-01-09T22:08:39.743-0800 I CONTROL  [initandlisten] MongoDB starting : pid=42675 port=27017 dbpath=/data/db 64-bit host
=ML-ITS-603436
2018-01-09T22:08:39.744-0800 I CONTROL  [initandlisten] db version v3.2.10
2018-01-09T22:08:39.744-0800 I CONTROL  [initandlisten] git version: 79d9b3ab5ce20f51c272b4411202710a082d0317
2018-01-09T22:08:39.744-0800 I CONTROL  [initandlisten] OpenSSL version: OpenSSL 1.0.2j  26 Sep 2016
2018-01-09T22:08:39.744-0800 I CONTROL  [initandlisten] allocator: system
2018-01-09T22:08:39.744-0800 I CONTROL  [initandlisten] modules: none
2018-01-09T22:08:39.744-0800 I CONTROL  [initandlisten] build environment:
2018-01-09T22:08:39.744-0800 I CONTROL  [initandlisten]     distarch: x86_64
2018-01-09T22:08:39.744-0800 I CONTROL  [initandlisten]     target_arch: x86_64
2018-01-09T22:08:39.744-0800 I CONTROL  [initandlisten] options: {}
2018-01-09T22:08:39.748-0800 I -        [initandlisten] Detected data files in /data/db created by the 'wiredTiger' storage
 engine, so setting the active storage engine to 'wiredTiger'.
2018-01-09T22:08:39.748-0800 I STORAGE  [initandlisten] wiredtiger_open config: create,cache_size=9G,session_max=20000,evic
tion=(threads_max=4),config_base=false,statistics=(fast),log=(enabled=true,archive=true,path=journal,compressor=snappy),fil
e_manager=(close_idle_time=100000),checkpoint=(wait=60,log_size=2GB),statistics_log=(wait=0),
2018-01-09T22:08:40.960-0800 I CONTROL  [initandlisten]
2018-01-09T22:08:40.960-0800 I CONTROL  [initandlisten] ** WARNING: soft rlimits too low. Number of files is 256, should be
 at least 1000
2018-01-09T22:08:41.070-0800 I NETWORK  [HostnameCanonicalizationWorker] Starting hostname canonicalization worker
2018-01-09T22:08:41.070-0800 I FTDC     [initandlisten] Initializing full-time diagnostic data capture with directory '/dat
a/db/diagnostic.data'
2018-01-09T22:08:41.072-0800 W NETWORK  [HostnameCanonicalizationWorker] Failed to obtain address information for hostname
ML-ITS-603436: nodename nor servname provided, or not known
2018-01-09T22:08:41.110-0800 I NETWORK  [initandlisten] waiting for connections on port 27017
2018-01-09T22:08:42.058-0800 I NETWORK  [initandlisten] connection accepted from 127.0.0.1:57341 #1 (1 connection now open)
```

Load Data from MongoDB to Spark

Provide the mongo-spark-connector package as a packages parameter value.

```
pyspark --packages org.mongodb.spark:mongo-spark-connector_2.11:2.2.0
```



https://docs.mongodb.com/spark-connector/master/

Load Data from MongoDB to Spark

Loading Data with SparkSQL

- MongoDB

```
df =
sqlContext.read.format("com.mongodb.spark.sql.DefaultSource")
    .option("uri","mongodb://127.0.0.1/msan698.sms").load()
```

Spark MLlib – spark_pam_spam_classification

Logistic Regression Example

- Spam Classification

1. Create an RDD.

```
from pyspark import SparkContext
from pyspark.sql.types import *
sc = SparkContext.getOrCreate()
sqlContext = SQLContext(sc)
```

```
df = sqlContext.read.format("com.mongodb.spark.sql.DefaultSource")\
    .option("uri","mongodb://127.0.0.1/msan698.sms").load()
```

```
ham_df = df.filter(df['Label']=='ham')
```

```
spam_df = df.filter(df['Label']=='spam')
```

Spark MLlib – spark_pam_spam_classification

Logistic Regression Example

- Spam Classification

1. Create an RDD.

```
ham_df.show()
```

```
+-----+--------------------+--------------------+
|Label|             Message|                 _id|
+-----+--------------------+--------------------+
|  ham|Go until jurong p...|[5aa9a8965c50bf0b...|
|  ham|I'm gonna be home...|[5aa9a8965c50bf0b...|
|  ham|U dun say so earl...|[5aa9a8965c50bf0b...|
|  ham|Oh k...i'm watchi...|[5aa9a8965c50bf0b...|
|  ham|Eh u remember how...|[5aa9a8965c50bf0b...|
|  ham|Ok lar... Joking ...|[5aa9a8965c50bf0b...|
|  ham|Fine if thats th...|[5aa9a8965c50bf0b...|
|  ham|Is that seriously...|[5aa9a8965c50bf0b...|
|  ham|I'm going to try ...|[5aa9a8965c50bf0b...|
|  ham|So ü pay first la...|[5aa9a8965c50bf0b...|
|  ham|Aft i finish my l...|[5aa9a8965c50bf0b...|
|  ham|Fffffffff. Alrig...|[5aa9a8965c50bf0b...|
|  ham|I HAVE A DATE ON ...|[5aa9a8965c50bf0b...|
|  ham|As per your reque...|[5aa9a8965c50bf0b...|
|  ham|I've been searchi...|[5aa9a8965c50bf0b...|
|  ham|Nah I don't think...|[5aa9a8965c50bf0b...|
|  ham|Lol your always s...|[5aa9a8965c50bf0b...|
|  ham|Did you catch the...|[5aa9a8965c50bf0b...|
|  ham|I'm back &amp;amp; we...|[5aa9a8965c50bf0b...|
|  ham|Ahhh. Work. I vag...|[5aa9a8965c50bf0b...|
+-----+--------------------+--------------------+
only showing top 20 rows
```

Spark MLlib – spark_pam_spam_classification

Logistic Regression Example

- Spam Classification

1. Create an RDD.

```
spam_df.show()
```

```
+-----+--------------------+--------------------+
|Label|             Message|                 _id|
+-----+--------------------+--------------------+
| spam|WINNER!! As a val...|[5aa9a8965c50bf0b...|
| spam|Had your mobile 1...|[5aa9a8965c50bf0b...|
| spam|SIX chances to wi...|[5aa9a8965c50bf0b...|
| spam|URGENT! You have ...|[5aa9a8965c50bf0b...|
| spam|Free entry in 2 a...|[5aa9a8965c50bf0b...|
| spam|XXXMobileMovieClu...|[5aa9a8965c50bf0b...|
| spam|England v Macedon...|[5aa9a8965c50bf0b...|
| spam|FreeMsg Hey there...|[5aa9a8965c50bf0b...|
| spam|Thanks for your s...|[5aa9a8965c50bf0b...|
| spam|07732584351 - Rod...|[5aa9a8965c50bf0b...|
| spam|SMS. ac Sptv: The...|[5aa9a8965c50bf0b...|
| spam|As a valued custo...|[5aa9a8965c50bf0b...|
| spam|Urgent UR awarded...|[5aa9a8965c50bf0b...|
| spam|Did you hear abou...|[5aa9a8965c50bf0b...|
| spam|Congrats! 1 year ...|[5aa9a8965c50bf0b...|
| spam|Your free rington...|[5aa9a8965c50bf0b...|
| spam|Please call our c...|[5aa9a8965c50bf0b...|
| spam|GENT! We are tryi...|[5aa9a8965c50bf0b...|
| spam|You are a winner ...|[5aa9a8965c50bf0b...|
| spam|PRIVATE! Your 200...|[5aa9a8965c50bf0b...|
+-----+--------------------+--------------------+
only showing top 20 rows
```

Spark MLlib – spark_pam_spam_classification

Logistic Regression Example

- Spam Classification

1. Create an RDD.

```
ham_rdd = ham_df.rdd.map(lambda x : x['Message'])
```

```
spam_rdd = spam_df.rdd.map(lambda x : x['Message'])
```

Spark MLlib – spark_pam_spam_classification

Logistic Regression Example

- Spam Classification

2. Clean the data.

Feature Extraction/Transformation.

- `HashingTF`
  - Maps a sequence of terms to their term frequencies using the hashing trick.

https://spark.apache.org/docs/1.2.0/api/scala/index.html#org.apache.spark.mllib.feature.HashingTF

Spark MLlib – spark_pam_spam_classification

Logistic Regression Example

- Spam Classification

2. Clean the data.

```
from pyspark.mllib.feature import HashingTF
tf = HashingTF(numFeatures = 100)
```

```
spamFeatures = spam_rdd.map(lambda x : tf.transform(x.split(" ")))
```

```
hamFeatures = ham_rdd.map(lambda x : tf.transform(x.split(" ")))
```

```
spamFeatures.collect()
```

```
[SparseVector(100, {0: 1.0, 4: 2.0, 8: 1.0, 20: 1.0, 21: 1.0, 22: 2.0, 24: 1.0, 27: 1.0, 37: 1.0, 40: 1.0, 43: 2.0,
51: 1.0, 56: 1.0, 62: 1.0, 65: 2.0, 70: 1.0, 72: 1.0, 82: 1.0, 84: 2.0, 88: 1.0, 89: 1.0}),
 SparseVector(100, {2: 1.0, 3: 1.0, 8: 2.0, 10: 1.0, 11: 1.0, 14: 1.0, 17: 1.0, 18: 1.0, 19: 1.0, 27: 1.0, 28: 1.0,
40: 1.0, 42: 1.0, 44: 1.0, 47: 1.0, 59: 1.0, 62: 1.0, 65: 2.0, 67: 1.0, 69: 2.0, 81: 1.0, 84: 2.0, 95: 2.0, 98: 1.
0}),
 SparseVector(100, {1: 2.0, 4: 1.0, 10: 1.0, 11: 1.0, 16: 1.0, 20: 1.0, 21: 1.0, 30: 1.0, 33: 2.0, 34: 1.0, 37: 1.
0, 48: 1.0, 52: 1.0, 55: 1.0, 62: 1.0, 63: 1.0, 65: 3.0, 66: 1.0, 84: 1.0, 86: 1.0, 93: 1.0, 97: 1.0}),
```

Spark MLlib – spark_pam_spam_classification

Logistic Regression Example

- Spam Classification

2. Clean the data.

`LabeldPoint (label, features)`

```
from pyspark.mllib.regression import LabeledPoint
positiveExamples = spamFeatures.map(lambda x : LabeledPoint(1, x))
negativeExamples = hamFeatures.map(lambda x : LabeledPoint(0, x))
training_data = positiveExamples.union(negativeExamples)
training_data.cache()
```

```
UnionRDD[246] at union at NativeMethodAccessorImpl.java:0
```

```
training_data.collect()
```

```
[LabeledPoint(1.0, (100,[0,4,8,20,21,22,24,27,37,40,43,51,56,62,65,70,72,82,84,88,89],[1.0,2.0,1.0,1.0,1.0,2.0,1.0,
1.0,1.0,1.0,2.0,1.0,1.0,1.0,2.0,1.0,1.0,1.0,2.0,1.0,1.0])),
 LabeledPoint(1.0, (100,[2,3,8,10,11,14,17,18,19,27,28,40,42,44,47,59,62,65,67,69,81,84,95,98],[1.0,1.0,2.0,1.0,1.
0,1.0,1.0,1.0,1.0,1.0,1.0,1.0,1.0,1.0,1.0,1.0,2.0,1.0,2.0,1.0,2.0,2.0,1.0])),
 LabeledPoint(1.0, (100,[1,4,10,11,16,20,21,30,33,34,37,48,52,55,62,63,65,66,84,86,93,97],[2.0,1.0,1.0,1.0,1.0,1.0,
1.0,1.0,2.0,1.0,1.0,1.0,1.0,1.0,1.0,1.0,3.0,1.0,1.0,1.0,1.0,1.0])),
```

https://spark.apache.org/docs/0.8.1/api/mllib/org/apache/spark/mllib/regression/LabeledPoint.html

Spark MLlib – spark_pam_spam_classification

Logistic Regression Example

- Spam Classification

3. Train the model.

```
from pyspark.mllib.classification import LogisticRegressionWithSGD
```

```
model = LogisticRegressionWithSGD.train(training_data)
```

```
/usr/local/Cellar/apache-spark/2.2.0/libexec/python/pyspark/mllib/classification.py:313: UserWarning: Deprecated in
2.0.0. Use ml.classification.LogisticRegression or LogisticRegressionWithLBFGS.
  "Deprecated in 2.0.0. Use ml.classification.LogisticRegression or "
```

Spark MLlib – spark_pam_spam_classification

Logistic Regression Example

- Spam Classification

3. Train the model. (check the created model.)

```
#Interpret the model
import operator
print(",".join([str(s) for s in sorted(enumerate([abs(x) for x in model.weights.toArray()]), key=operator.itemgetter(0)
```

```
(0, 0.07291910167120795),(1, 0.27641028312863813),(2, 0.180843065044175 52),(3, 0.10776168428967071),(4, 0.54050102526
986021),(5, 0.0),(6, 0.13332249583574537),(7, 0.0),(8, 0.67927755040357385),(9, 0.076523781524273499),(10, 0.0),(11,
0.008754005119169973),(12, 0.02702461938482734),(13, 0.0),(14, 0.19206934808011311),(15, 0.0),(16, 0.3732377830870003
4),(17, 0.18651434450400761),(18, 0.078567673616367364),(19, 0.026853653187871338),(20, 0.078567673616367364),(21, 0.
19609183310322334),(22, 0.0),(23, 0.2248672288970984),(24, 0.12061838368254736),(25, 0.2334900860404964),(26, 0.19948
462262476277),(27, 0.0),(28, 0.11843820447040838),(29, 0.41745598205794476),(30, 0.0),(31, 0.067639955906277319),(32,
 0.0),(33, 0.42159583838964215),(34, 0.33965904422936471),(35, 0.0),(36, 0.11735674213677989),(37, 0.0993122390704887
5),(38, 0.12389870038959433),(39, 0.077475423359075785),(40, 0.10328743243450288),(41, 0.0),(42, 0.27063702169648068)
,(43, 0.45780311102801352),(44, 0.30299871838614351),(45, 0.11623413410502156),(46, 0.26768663578393503),(47, 0.0),(4
8, 0.18607816037668568),(49, 0.10751048676031848),(50, 0.57763213108156231),(51, 0.033566577637641808),(52, 0.1920693
4808011311),(53, 0.15495084671815157),(54, 0.0),(55, 0.21079791919482108),(56, 0.11735674213677989),(57, 0.2877914945
5159897),(58, 0.26700021187949907),(59, 0.66996561350538286),(60, 0.21079791919482108),(61, 0.15948972511918019),(62,
```

Spark MLlib – spark_pam_spam_classification

Logistic Regression Example

- Spam Classification

3. Train the model. (Save the model.)

```
#  save the model
model.save(sc, "logistic_regression_model.model")
```

Spark MLlib – spark_pam_spam_classification

Logistic Regression Example

- Spam Classification

4. Test/Evaluate (Load the model).

```
#  Test the model.

#  load the model
from pyspark import SparkContext
from pyspark.sql.types import *
from pyspark.mllib.classification import LogisticRegressionModel
from pyspark.mllib.feature import HashingTF

sc = SparkContext.getOrCreate()
sqlContext = SQLContext(sc)
sqlContext.read.format("com.mongodb.spark.sql.DefaultSource").option("uri","mongodb://127.0.0.1/msan698.sms").load()
tf = HashingTF(numFeatures = 100)

loaded_model = LogisticRegressionModel.load(sc, "logistic_regression_model.model")
```

Spark MLlib – spark_pam_spam_classification

Logistic Regression Example

- Spam Classification

4. Test/Evaluate.

You can try with more test data sets and RegressionMetrics to calculate MSE/RMSE.

```
posTestExample = tf.transform("Text me for a FREE iPhoneX!".split(" "))
negTestExample = tf.transform("Good luck with your job search!".split(" "))
```

```
print ("Prediction for positive test example : ", loaded_model.predict(posTestExample))
print ("Prediction for negative test example : ", loaded_model.predict(negTestExample))
```

```
Prediction for positive test example :  1
Prediction for negative test example :  0
```

Reference

Flask SQLAlchemy, http://flask-sqlalchemy.pocoo.org/2.3/

Werkzeug Security Helpers, http://werkzeug.pocoo.org/docs/0.14/utils/#module-werkzeug.security

Flask Login, https://flask-login.readthedocs.io

Kenneth Reitz. *The Hitchhiker's Guide to Python*. http://docs.python-guide.org/en/latest/

Grinberg, Miguel. *Flask web development: developing web applications with python*. " O'Reilly Media, Inc.", 2018.