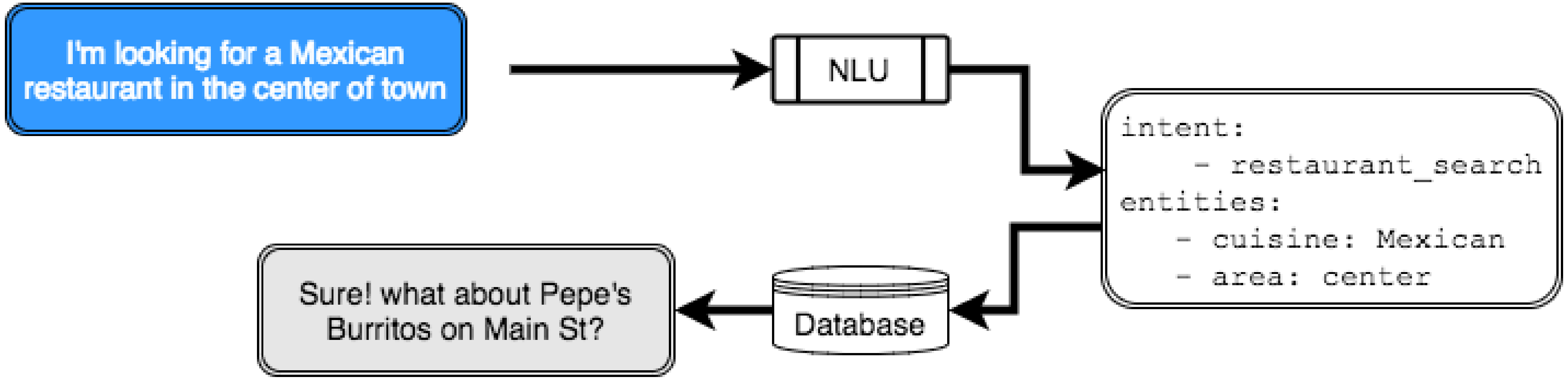
Understanding intents and entities

BUILDING CHATBOTS IN PYTHON

Alan Nichol
Co-founder and CTO, Rasa

An example



I'm looking for a Mexican restaurant in the center of town

NLU

```
intent:
    - restaurant_search
entities:
   - cuisine: Mexican
   - area: center
```

Database

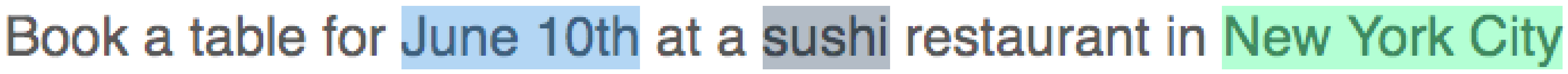
Sure! what about Pepe's Burritos on Main St?

Intents

A `restaurant_search` can be expressed many different ways:

- I'm hungry
- Show me good pizza spots
- I want to take my boyfriend out for sushi
    - Can also be `request_booking`

Entities



Book a table for June 10th at a sushi restaurant in New York City

- NER = Named Entity Recognition

Regular expressions to recognize intents

- Simpler than machine learning approaches
- Highly computationally efficient
- Drawback:
  - Debugging regular expressions can become difficult

Using regular expressions

- `'|'` is equivalent to OR

```
re.search(r"(hello|hey|hi)", "hey there!") is not None
```

```
True
```

```
re.search(r"(hello|hey|hi)", "which one?") is not None
```

```
True
```

Using regular expressions

- `\b` matches the beginning or end of a word

```
re.search(r"\b(hello|hey|hi)\b", "hey there!") is not None
```

```
True
```

```
re.search(r"\b(hello|hey|hi)\b", "which one?") is not None
```

```
False
```

Using regex for entity recognition

```
pattern = re.compile('[A-Z]{1}[a-z]*')
message = """
Mary is a friend of mine,
she studied at Oxford and
now works at Google"""
pattern.findall(message)
```

```
['Mary', 'Oxford', 'Google']
```

Let's practice!

BUILDING CHATBOTS IN PYTHON

Word vectors

BUILDING CHATBOTS IN PYTHON

Alan Nichol
Co-founder and CTO, Rasa

Machine learning

- Programs which can get better at a task by being exposed to more data
- Identifying which intent a user message belongs to

Vector representations

"can you help me please?"

Units	examples	vectors
characters	`"c", "a", "n",...`	`v_c, v_a, v_n, ...`
words	`"can", "you", ...`	`v_{can}, v_{you}, ...`
sentences	`"can you help..."`	`v_{can you help ...}`

Word vectors

Context	Candidates
let's meet at the ___ tomorrow	office, gym, park, beach, party
I love going to the ___ to play with the dogs	beach, park

- Word vectors try to represent *meaning* of words
- Words which appear in similar context have similar vectors

Word vectors are computationally intensive

- Training word vectors requires a lot of data
- High quality word vectors are available for anyone to use
- GloVe algorithm
  - Cousin of word2vec
- spaCy

```python
import spacy
nlp = spacy.load('en')
nlp.vocab.vectors_length
```

```
300
```

```python
doc = nlp('hello can you help me?')
for token in doc:
    print("{} : {}".format(token, token.vector[:3]))
```

```
hello : [ 0.25233001  0.10176    -0.67484999]
can : [-0.23857     0.35457    -0.30219001]
you : [-0.11076     0.30785999 -0.51980001]
help : [-0.29370001  0.32253    -0.44779    ]
me : [-0.15396     0.31894001 -0.54887998]
? : [-0.086864    0.19160999  0.10915   ]
```

Similarity

- Direction of vectors matters
- "Distance" between words = angle between the vectors
- Cosine similarity
  - 1: If vectors point in the same direction
  - 0: If they are perpendicular
  - -1: If they point in opposite directions

.similarity()

- "can" and "cat" are spelled similarly but have low similarity
- but "cat" and "dog" have high similarity

```
import spacy
nlp = spacy.load('en')
doc = nlp("cat")
doc.similarity(nlp("can"))
```

```
0.30165292161215396
```

```
doc.similarity(nlp("dog"))
```

```
0.8016855173294953
```

Let's practice!

Intents and classification

BUILDING CHATBOTS IN PYTHON

Alan Nichol
Co-founder and CTO, Rasa

Supervised learning

- A classifier predicts the intent label given a sentence
- "Fit" classifier by tuning it on *training data*
- Evaluate performance on *test data*
- Accuracy: the fraction of labels we predict correctly

ATIS dataset

- Thousands of sentences with labeled intents and entities
- Collected from a real flight booking service
- Intents like
    - `atis_flight`
    - `atis_airfare`

ATIS dataset II

```python
sentences_train[:2]
labels_train[:2]
```

```
[ "atis_flight",
  "atis_flight"]
```

```
["i want to fly from boston at
  838 am and arrive in denver at
  1110 in the morning",
  "what flights are available
  from pittsburgh to baltimore
  on thursday morning"]
```

```python
import numpy as np
X_train_shape = (
    len(sentences_train),
    nlp.vocab.vectors_length)
X_train = np.zeros(X_train_shape)
for sentence in sentences_train:
    X_train[i,:] = nlp(sentence).vector
```

Nearest neighbor classification

- Need training data
  - Sentences which we've already labeled with their intents
- Simplest solution:
  - Look for the labeled example that's most similar
  - Use its intent as a best guess
- Nearest neighbor classification

Nearest neighbor classification in scikit-learn

```
from sklearn.metrics.pairwise import cosine_similarity
test_message = """
i would like to find a flight from charlotte
to las vegas that makes a stop in st. louis"""
test_x = nlp(test_message).vector
scores = [
    cosine_similarity(X[i,:], test_x)
    for i in range(len(sentences_train)
]
labels_train[np.argmax(scores)]
```

```
'atis_flight'
```

Support vector machines

- Nearest neighbors is very simple - we can do better
- SVM / SVC: support vector machine / classifier

```python
from sklearn.svm import SVC
clf = SVC()
clf.fit(X_train, y_train)
y_pred = clf.predict(X_test)
```

Let's practice!

BUILDING CHATBOTS IN PYTHON

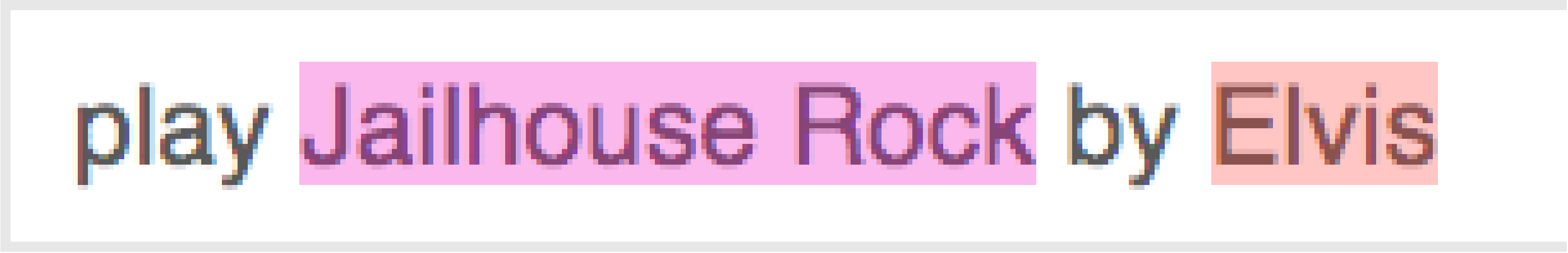
Entity extraction

BUILDING CHATBOTS IN PYTHON

Alan Nichol

Co-founder and CTO, Rasa

Beyond keywords: context



play Jailhouse Rock by Elvis

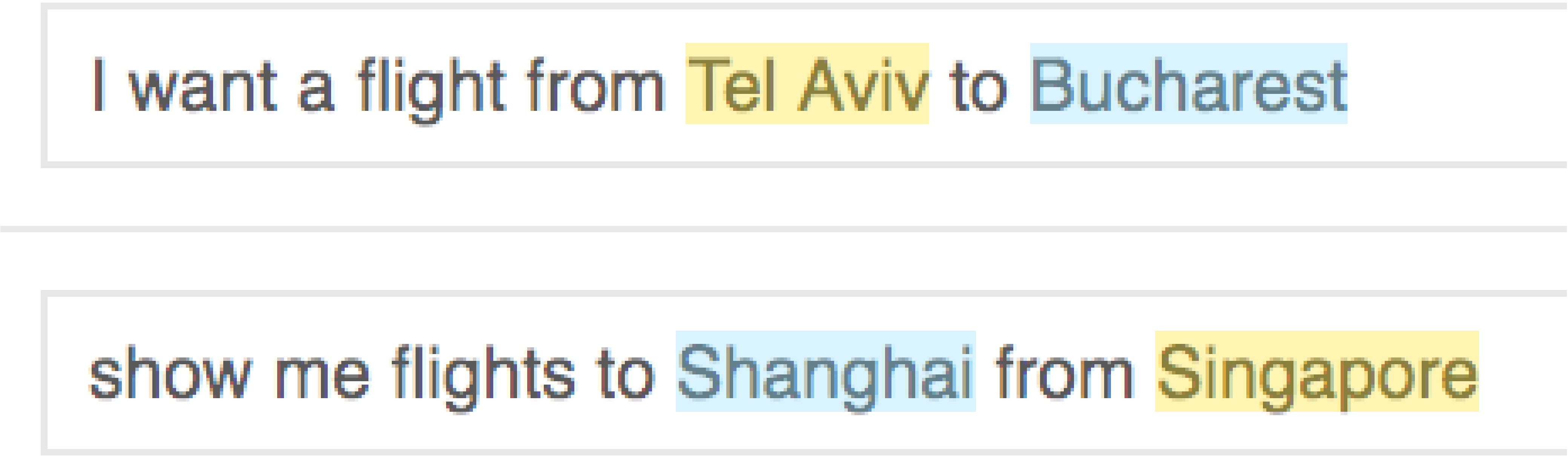
- Keywords don't work for entities you haven't seen before
- Use contextual clues:
  - Spelling
  - Capitalization
  - Words occurring before & after
- Pattern recognition

Pre-built Named Entity Recognition

```python
import spacy
nlp = spacy.load('en')
doc = nlp("my friend Mary has worked at Google since 2009")
for ent in doc.ents:
    print(ent.text, ent.label_)
```

```
Mary PERSON
Google ORG
2009 DATE
```

Roles

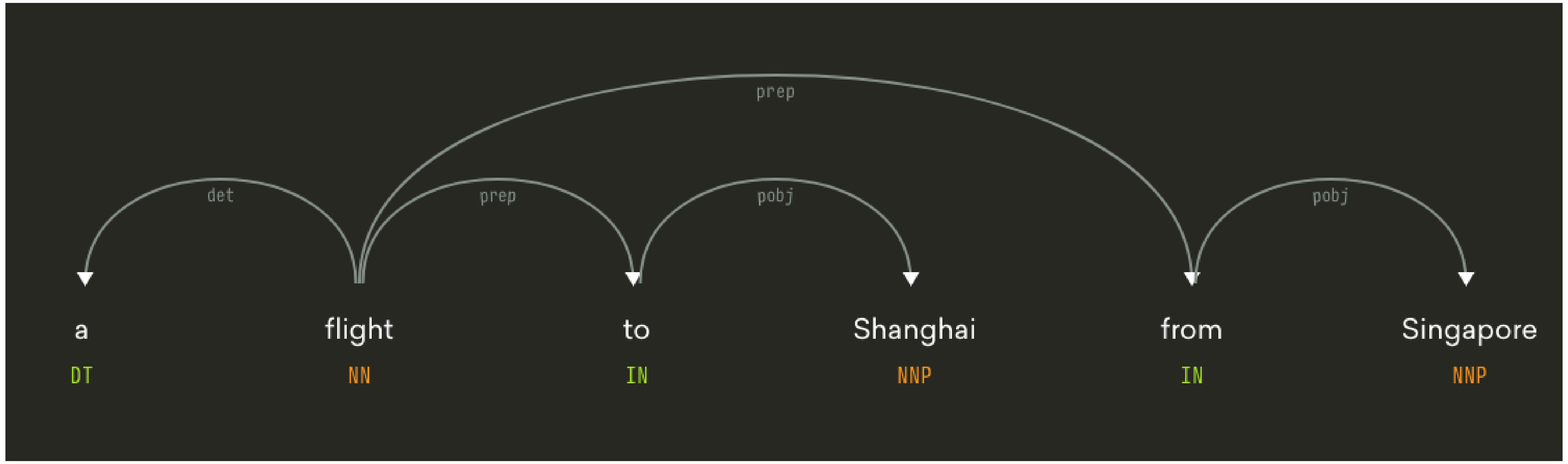


I want a flight from Tel Aviv to Bucharest

show me flights to Shanghai from Singapore

```
pattern_1 = re.compile('.* from (.*) to (.*)')

pattern_2 = re.compile('.* to (.*) from (.*)')
```



```
doc = nlp('a flight to Shanghai from Singapore')
shanghai, singapore = doc[3], doc[5]
list(shanghai.ancestors)
```

```
[to, flight]
```

```
list(singapore.ancestors)
```

```
[from, flight]
```

Shopping example

```python
doc = nlp("let's see that jacket in red and some blue jeans")
items = [doc[4], doc[10]]  # [jacket, jeans]

colors = [doc[6], doc[9]]  # [red, blue]
for color in colors:
    for tok in color.ancestors:
        if tok in items:
            print("color {} belongs to item {}".format(color, tok))
            break
```

```
color red belongs to item jacket
color blue belongs to item jeans
```

Let's practice!

Robust NLU with Rasa

BUILDING CHATBOTS IN PYTHON

Alan Nichol
Co-founder and CTO, Rasa

Rasa NLU

- Library for intent recognition & entity extraction
- Based on spaCy, scikit-learn, & other libraries
- Built in support for chatbot specific tasks

Rasa data format

```python
from rasa_nlu.converters import load_data
training_data = load_data("./training_data.json")
import json
print(json.dumps(data.training_examples[22], indent=2))
```

```
{ "text": "i'm looking for a place in the north of town",
  "intent": "restaurant_search",
  "entities": [
    { "start": 31,
      "end": 36,
      "value": "north",
      "entity": "location" }
  ]
}
```

Interpreters

```
message = "I want to book a flight to London"
interpreter.parse(message))
```

```
{ "intent": {
    "name": "flight_search",
    "confidence": 0.9
  },
  "entities": [
    { "entity": "location",
      "value": "London",
      "start": 27,
      "end": 33
    }
  ]
}
```

Rasa usage

```
# Creating a model
from rasa_nlu.config import RasaNLUConfig
from rasa_nlu.model import Trainer
config = RasaNLUConfig(
            cmdline_args={"pipeline": "spacy_sklearn"})
trainer = Trainer(config)
interpreter = trainer.train(training_data)
```

Rasa pipelines

```
spacy_sklearn_pipeline = [
  "nlp_spacy",
  "ner_crf",
  "ner_synonyms",
  "intent_featurizer_spacy",
  "intent_classifier_sklearn"  ]
# These two statements are identical:
RasaNLUConfig(cmdline_args={"pipeline": spacy_sklearn_pipeline})
```

```
<rasa_nlu.config.RasaNLUConfig at 0x10f60aa90>
```

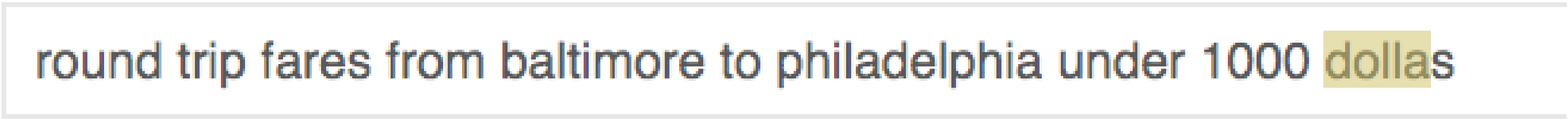
```
RasaNLUConfig(cmdline_args={"pipeline": "spacy_sklearn"})
```

```
<rasa_nlu.config.RasaNLUConfig at 0x10f60aa20>
```

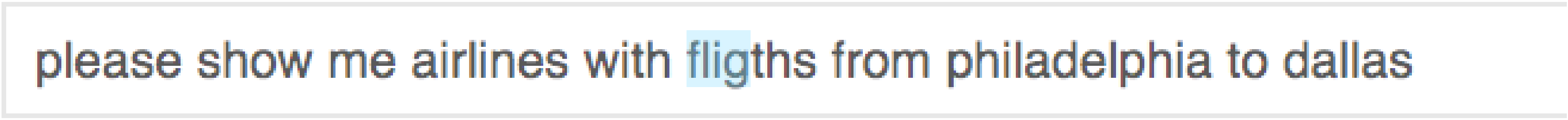
Conditional random fields

- Machine Learning model, popular for named entity recognition
  - can perform well even with small training data

Handling typos



round trip fares from baltimore to philadelphia under 1000 dollas



please show me airlines with fligths from philadelphia to dallas

```
pipeline = [
    "nlp_spacy",
    "intent_featurizer_spacy",
    "intent_featurizer_ngrams",
    "intent_classifier_sklearn"
]
```

Let's practice!

BUILDING CHATBOTS IN PYTHON